INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA

SECCIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

IMPLEMENTACIÓN DE UN MEDIDOR CON MICROCONTROLADOR PARA DETECCIÓN DE OSCILACIONES DE BAJA FRECUENCIA

TESIS

QUE PARA OBTENER EL GRADO DE:
MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

PRESENTA:

SALVADOR DAVID GARCÍA VEGA

Ciudad de México 2011

SIP-14-BIS

INSTITUTO POLITÉCNICO NACIONAL

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

ACTA DE REVISIÓN DE TESIS

En la Ciudad de MÉXICO D. F. siendo las 17:00 horas del día 7 del mes de Diciembre del 2011 se reunieron los miembros de la Comisión Revisora de la Tesis, designada por el Colegio de Profesores de Estudios de Posgrado e Investigación de la **E. S. I. M. E. ZAC.** para examinar la tesis titulada:

"IMPLEMENTACIÓN DE UN MEDIDOR CON MICROCONTROLADOR PARA DETECCIÓN DE OSCILACIONES DE BAJA FRECUENCIA"

Presentada por el alumno:

GARCÍA	VEGA	SALVADOR DAVID
Apellido paterno	Apellido materno	Nombre(s)

Con registro: B 0 9 1 6 1 4

aspirante de:

MAESTRÍA EN CIENCIAS EN INGENIERÍA ELÉCTRICA

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

LA COMISIÓN REVISORA

Directores de tesis

DR. DAVID ROMERO ROMERO

DR. RAÚL ÁNGEL CORTÉS MATEOS

DR. DAVID SEBASTIÁN BALTAZAR
Presidente

DR. DANIEL RUIZ VEGA
Secretario

DR. GERMÁN ROSAS ORTIZ
Tercer vocal

PRESIDENTE DEL COLEGIO DE PROFESORES

ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA
ESTADOS UNIDOS MEXICANOS
I. P. N.
DIRECCIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

DR. JAIME ROBLES GARCÍA

INSTITUTO POLITÉCNICO NACIONAL

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

CARTA CESIÓN DE DERECHOS

En la Ciudad de México, Distrito Federal, el día 7 del mes de Diciembre del año 2011, el (la) que suscribe **Salvador David García Vega** alumno (a) del Programa de Maestría en Ciencias en Ingeniería Eléctrica con número de registro B091614, 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. **David Romero Romero** y del Dr. **Raúl Ángel Cortés Mateos** y cede los derechos del trabajo intitulado **"Implementación de un medidor con microcontrolador para detección de oscilaciones de baja frecuencia"**, al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

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

Salvador David García Vega

AGRADECIMIENTOS

A mis asesores el Dr. Raúl Cortés Mateos y el Dr. David Romero Romero por la dirección brindada a mi trabajo de investigación, mi formación profesional y personal, sus comentarios y sugerencias fueron muy valiosos para la finalización de este trabajo.

A la comisión revisora (Dr. David Sebastián Baltazar, Dr. Daniel Ruiz Vega, Dr. German Rosas Ortiz y Dr. Domitilio Libreros) por sus valiosas aportaciones en la mejora de este trabajo

Al Ing. Miguel Ángel Álvarez Juárez por el apoyo brindado en la realización de las simulaciones en el programa DSA Tools y el uso del simulador en tiempo real.

Al IPN por proporcionarme la beca tesis que ayudo para la finalización de este proyecto.

Este trabajo fue realizado con el apoyo económico bridado por el Consejo Nacional de Ciencia y Tecnología (CONACYT).

DEDICATORIAS

A mis padres David García Baños y Margarita Vega López

A mis hermanos Melisa y Sandino

A Sayira Rojo Santiesteban

A todos mis amigos

RESUMEN

En este trabajo se presenta una implementación utilizando un microcontrolador, que a partir de mediciones fasoriales, calcula la potencia activa, potencia reactiva y magnitud de voltaje enviando estas variables a través de la interfaz RS-232 a un programa de detección de oscilaciones de baja frecuencia programado en MATLAB.

En la programación del microcontrolador se utiliza un sistema operativo en tiempo real para la medición de los fasores de voltaje y corriente, cálculo de la magnitud de voltaje, potencia activa, potencia reactiva y envío de datos al programa en MATLAB.

El programa en MATLAB realiza la detección de eigenvalores oscilatorios de baja frecuencia utilizando el algoritmo de Prony como herramienta. El programa inicia con el código de la interfaz que recibe los datos enviados desde el microcontrolador, en donde se prepara el vector de entrada para el algoritmo de Prony, que identifica los modos oscilatorios de la señal de entrada.

El prototipo se prueba mediante un sistema eléctrico de potencia de dos áreas con oscilaciones de 0.5 a 3 Hz, en un simulador de sistemas de potencia en tiempo real.

Con la identificación modal por el método de Prony, se detectaron las oscilaciones de baja frecuencia permitiendo corroborar los resultados obtenidos con los existentes en la literatura.

De los resultados obtenidos se observa que el código desarrollado para el algoritmo de Prony permite detectar oscilaciones de baja frecuencia ante diferentes valores de entrada.

ABSTRACT

This thesis presents an implementation using a microcontroller that measures and calculates; Voltage and current phasors, active power, reactive power and voltage magnitude. These variables are sent via an RS-232 interface to a MATLAB program for detecting low frequency oscillations.

The microcontroller software was made using a real-time operating system to measure voltage and current phasors, calculation of active and reactive power and data communication to a PC MATLAB program.

The MATLAB program detects low-frequency oscillatory eigenvalues using the Prony algorithm. The program receives data from the microcontroller and prepares the input vector for the Prony method that identifies oscillatory modes of the input signal.

The prototype is tested in a real-time simulator, using a two area electric power system with oscillations between 0.5 to 3 Hz.

The modal identification detected low-frequency oscillations and corroborated results obtained in the literature.

The obtained results show that the developed program of the Prony method can detect low-frequency oscillations.

CONTENIDO

LISTA DE FIGURAS

Página

Página

Página

LISTA DE TABLAS

Página

NOMENCLATURA

$\boldsymbol{\lambda_i}$	i-ésimo eigenvalor
$\boldsymbol{\zeta}$	Relación de amortiguamiento
$\boldsymbol{\sigma}$	Factor de amortiguamiento
$\boldsymbol{h_k}$	Amplitud compleja
$\boldsymbol{z_k}$	Exponencial compleja
$\boldsymbol{A_k}$	Amplitud de la exponencial compleja
$\boldsymbol{p}$	Número de exponenciales complejas
$\boldsymbol{\theta_k}$	Fase relativa
$\boldsymbol{f_k}$	Frecuencia de oscilación
$\boldsymbol{T}$	Periodo
$\boldsymbol{\varphi(z)}$	Polinomio de aproximación
$\boldsymbol{a(m)}$	Coeficientes constantes del polinomio de aproximación
$\boldsymbol{x(n)}$	Muestra
$\boldsymbol{\hat{x}(n)}$	Aproximación de la señal de entrada del método de Prony
$\boldsymbol{R_i}$	iésima matriz residual
$\boldsymbol{A}$	Matriz de estado de dimensión n x n
$\boldsymbol{V}$	Magnitud de voltaje
$\boldsymbol{\partial}$	Ángulo del fasor de voltaje
$\boldsymbol{P_G}$	Potencia activa de generación
$\boldsymbol{Q_G}$	Potencia reactiva de generación
$\boldsymbol{P_d}$	Potencia activa demandada
$\boldsymbol{Q_d}$	Potencia reactiva demandada
$\boldsymbol{H}$	Constante de inercia
$\boldsymbol{D}$	Coeficiente de amortiguamiento
$\boldsymbol{R_a}$	Resistencia del estator
$\boldsymbol{X_l}$	Reactancia de fuga
$\boldsymbol{X_d}$	Reactancia síncrona en el eje directo
$\boldsymbol{X_q}$	Reactancia síncrona en el eje en cuadratura
$\boldsymbol{X_d^{'}}$	Reactancia transitoria en el eje directo.
$\boldsymbol{X_q^{'}}$	Reactancia transitoria en el eje en cuadratura.
$\boldsymbol{T_{d0}^{'}}$	Constante de tiempo transitoria de circuito abierto en el eje directo.
$\boldsymbol{T_{q0}^{'}}$	Constante de tiempo transitoria de circuito abierto en el eje en cuadratura.
$\boldsymbol{X_d^{''}}$	Reactancia subtransitoria en el eje directo
$\boldsymbol{X_q^{''}}$	Reactancia subtransitoria en el eje en cuadratura
$\boldsymbol{T_{d0}^{''}}$	Constante de tiempo subtransitoria de circuito abierto en el eje directo
$\boldsymbol{T_{q0}^{''}}$	Constante de tiempo subtransitoria de circuito abierto en el eje en cuadratura
$\boldsymbol{ReX}$	Parte real de vector de X
$\boldsymbol{ImX}$	Parte imaginaria del vector X

ABREVIATURAS

ADC	*"Analog-to-Digital Converter"* por sus siglas en inglés.
AVR	*"Automatic Voltage Regulator"* por sus siglas en inglés.
CA	Corriente Alterna.
CAN	*"Controller Area Network"* por sus siglas en inglés.
CD	Corriente Directa.
CAU	*"Cryptography Acceleration Unit"* por sus siglas en inglés.
CEV	Compensador Estático de Vars.
CFM	*"Coldfire Flash Module"* por sus siglas en inglés.
CIGRE	"*Conseil International des Grandes Reseaux Électriques*" en francés.
DFT	*"Discrete Fourier Transform"* por sus siglas en inglés.
DMA	"Direct Memory Access" por sus siglas en Inglés.
DSAT	"*Dynamic Security Assessment Tools*" por sus siglas en inglés.
FEC	*"Fast Ethernet controller"* por sus siglas en inglés.
FIFO	"First in, first out"en inglés.
GPIO	*"General Purpose I/O Timing"* por sus siglas en inglés.
GPS	*"Global Positioning System"* por sus siglas en inglés.
HVDC	*"High Voltage Direct Current*" por sus siglas en inglés.
NVM	"Non-Volatile Memory" por sus siglas en inglés.
PMU	"Phasor Measurement Unit" por sus siglas en inglés.
PSAT	"Power System Analysis Toolbox" en ingles.
PSS	"Power system Stabilizers" por sus siglas en inglés.
PSS/E	"*Power System Simulator* "por sus siglas en inglés.
PWM	"Pulse Width Modulation" por sus siglas en inglés.
QSPI	"Queued serial peripheral interface" por sus siglas en inglés.
RMS	"Root Mean Square" por sus siglas en inglés.
SEP	Sistema Eléctrico de Potencia
SNR	"*Signal to noise ratio*" por sus siglas en inglés.
SSAT	*"Small Signal Analysis Tool"* por sus siglas en inglés.
UART	*"Universal Asynchronous/synchronous Receiver Transmit"* en inglés
USB	*"Universal Serial Bus"* por sus siglas en Inglés.
WECC	"*Western Electricity Coordinating Council"* por sus siglas en inglés.

CAPÍTULO 1

INTRODUCCIÓN

En los sistemas eléctricos de potencia (SEP) existen ocasionalmente oscilaciones que pueden ser dañinas, debidas al comportamiento dinámico de estos.

En el presente trabajo se implementa un prototipo para detectar uno de los casos de oscilaciones mal amortiguadas, el de las oscilaciones de baja frecuencia, las cuales pueden ser detectadas por el método de Prony que es una de las técnicas más utilizadas debido a la facilidad que presenta en su formulación matemática.

El desarrollo del sistema se realizó mediante un microcontrolador que mide fasores y calcula la magnitud de voltaje, potencia activa y reactiva de un nodo de la red. Los valores calculados se transmiten cada 0.03 segundos a una Computadora Personal (PC) que mediante un Programa en MATLAB detecta oscilaciones de baja frecuencia cada 10 segundos.

1.1 OBJETIVO GENERAL

Realizar la detección de oscilaciones de baja frecuencia en un sistema de potencia en línea utilizando mediciones sincronizadas de fasores mediante un microcontrolador.

1.2 OBJETIVOS ESPECÍFICOS

- *Calcular mediante los fasores medidos de voltaje y corriente, la magnitud del voltaje, las potencias activa y reactiva de la señal obtenida de los nodos de un simulador de sistemas de potencia en tiempo real.*

- *Detectar las oscilaciones de baja frecuencia de la red utilizando el algoritmo de Prony.*

- *Mediante un simulador digital, analizar el comportamiento del método de Prony ante diferentes tipos de señales de entrada con oscilaciones de baja frecuencia.*

1.3 JUSTIFICACIÓN

Las oscilaciones mal amortiguadas de frecuencia en sistemas eléctricos de potencia representan un problema latente para las compañías encargadas del suministro de energía.
Actualmente, estas oscilaciones son provocadas debido a la operación de los SEP más cercanas a los límites de operación de los generadores, transformadores y líneas de transmisión, así como el uso de sistemas de excitación de respuesta rápida.
Por esta razón, es importante diseñar e implementar prototipos para detectar oscilaciones de baja frecuencia.

1.4 ALCANCES Y LIMITACIONES

En el microcontrolador se realizan mediciones de 30 fasores por segundo de cada señal medida por el ADC, que representa la velocidad más rápida obtenida con la programación y equipo utilizado en el presente trabajo.

Se presenta un programa de detección de oscilaciones de baja frecuencia que utiliza el análisis de Prony como método de identificación, este programa es probado ante diferentes señales de entrada.

Se presenta una implementación en línea utilizando un simulador digital de sistemas de potencia en tiempo real (OPAL-RT), interconectado al prototipo realizado.

1.5 ESTADO DEL ARTE

1.5.1 Trabajos relevantes acerca de las oscilaciones en sistemas de potencia

En Hauer *et al.* (1990) [1], se hace un breve resumen matemático del método de Prony, se mencionan algunas características importantes acerca del uso del método de Prony en estudios de estabilidad transitoria. Los resultados que se presentan para el análisis modal y la construcción de modelos detallados estan basados en datos de la respuesta obtenida a través de pruebas de gran escala del WECC.

En Hauer (1991) [2], se realiza una implementación del método de Prony en combinación con el análisis modal, se presenta la formulación matemática del método Prony, se describe un algoritmo para la programación del análisis de Prony.

En Grund *et al.* (1993) [3], se hace una comparación entre el análisis de eigenvalores y el análisis de Prony utilizando datos en el dominio de la frecuencia para la estimación del amortiguamiento de las oscilaciones presentes en un SEP, utilizando los resultados para el

diseño de controles para SEP. Además, este trabajo hace una corrección a la formulación de la relación señal ruido hecha en [1].

En el reporte del CIGRE (1996) [4], realiza una recopilación de los las oscilaciones de baja frecuencia presentadas en SEP alrededor del mundo, de este trabajo se utilizan las definiciones de los métodos de análisis de oscilaciones, la descripción del análisis modal y su formulación matemática y la clasificación de las oscilaciones en los sistemas de potencia.

En Trudnowski (1999) [5], se hace un análisis de varias señales en forma simultanea como extensión al método de Prony resultando mejor precisión en las estimaciones modales y la simplificación de los pasos del análisis. Se analiza la forma en que el ruido presente en una señal analizada afecta el proceso de identificación modal haciendo difícil el comparar los resultados obtenidos a partir de diferentes señales correspondientes a un mismo sistema. Para superar este problema, se propone extender el método de Prony para analizar varias señales al mismo tiempo

En Kaberere *et al*. (2005) [6], se presenta la experiencia adquirida en la comparación de una serie de herramientas de simulación del sistema eléctrico. Se llevó a cabo un análisis de eigenvalores del sistema de potencia de dos áreas-cuatro generadores con paqueterías comerciales.

En Ding *et al*. (2010) [7], se propone un método de identificación modal basado en el análisis de Prony, en donde se realiza el pre procesamiento del resultado del análisis de Prony, la identificación del modo predominante y la agrupación de los generadores; se proponen tres índices para detectar el modo dominante; el primero es la relación entre la amplitud y coeficiente de amortiguamiento; el segundo es el modo de energía y la tercero es la amplitud de modo de decaimiento sinusoidal después de un período de oscilación.

En Zhou *et al*. (2010) [8], se hace un estudio de oscilaciones electromecánicas utilizando el método de Prony usando datos de una unidad medidora fasorial (PMU en inglés), lleva a cabo una aplicación en línea para realizar las estimaciones modales del sistema de 17 nodos validando con simulaciones, demuestra que es capaz de realizar una detección de oscilaciones en línea utilizando el método de Prony.

En Liang *et al.* (2010) [9], se hace una modificación al método de Prony realizando ventanas de análisis recursivos para realizar la detección de eigenvalores oscilatorios. Además, se realiza un estudio de la propuesta con señales ideales sin ruido. Se utiliza en este trabajo el estudio con valores singulares del método de Prony.

1.5.2 Libros consultados

En libro de Scharf (1991) [10], en este documento se realiza la descripción original del método de Prony y las mejoras realizadas al método por otros autores, también se hacen descripciones del análisis de series de tiempo, utilización del método de detección y la estimación modal. En este libro se hace una descripción de las técnicas de análisis modal más utilizadas con su formulación matemática.
En el libro de Kundur (1994) [11], se obtuvieron conceptos relacionados con la estabilidad ante pequeños disturbios de un SEP; se obtienen de este libro la clasificación de las oscilaciones de baja frecuencia; la descripción de análisis de eigenvalores y resultados, y los datos para la simulación del sistema de dos áreas.

En el libro de Rogers (2000) [12], se obtienen conceptos acerca de la relación histórica de las oscilaciones de baja frecuencia con los sistemas eléctricos de potencia; la clasificación y algunos de los métodos que se utilizan para el análisis de oscilaciones; además, se mencionan los posibles motivos para la reaparición de las oscilaciones de baja frecuencia.

En el libro de Pai *et al*. (2005) [13], se realiza un estudio histórico importante de las oscilaciones de baja frecuencia en sistemas de potencia, también se hace una descripción de estabilidad a pequeños disturbios y la clasificación de los métodos de detección.

1.5.3 Tesis internacionales consultadas

En Wee Tan (2003) [14], Este proyecto describe el análisis de Prony y su aplicación a la evaluación de la estabilidad del sistema de potencia. Se utiliza el análisis de Prony para determinar el modo peculiar del modelo del sistema linealizado alrededor de un punto específico de operación.

1.5.4 Tesis desarrolladas en la SEPI-ESIME

En Reyes, 2005 [15] se presenta una alternativa para analizar la estabilidad del sistema eléctrico de potencia ante pequeños disturbios, el sistema de prueba es el sistema máquina síncrona bus-infinito y su estabilidad se determina utilizando una red neuronal. En este trabajo la determinación de la estabilidad se realiza como un reconocimiento de patrones; de esta manera la identificación de la estabilidad se puede realizar más fácil y eficientemente que mediante el cálculo y análisis de eigenvalores.

En Cuvas (2006) [16], se presenta la implementación de un medidor fasorial, sincronizado vía satélite mediante un módulo receptor GPS, se realizan mediciones de fasores de voltaje y corriente de un sistema trifásico a 60 HZ, los fasores calculados y el estampado de tiempo son enviados mediante un puerto de comunicación serial RS-232 a una PC. De esta tesis es importante destacar los algoritmos para la medición de fasores.

En Villarreal (2008) [17], se presentan los conceptos básicos del análisis de oscilaciones y del de análisis modal por el método QR. Adicionalmente, los resultados del análisis modal se compararon con los de programas de simulación en el tiempo que emplean al modelo no-lineal del sistema por medio del análisis de Prony.

En Hernández-Gómez (2009) [18], 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.

1.6 ESTRUCTURA DEL TRABAJO DE TESIS

En el capítulo 1, se describen los objetivos, la justificación, el alcance y las limitaciones, el estado del arte y la estructura de la tesis.

En el capítulo 2, se aborda de manera general el fenómeno de las oscilaciones de baja frecuencia en los sistemas de potencia, la clasificación de estas y los métodos que se utilizan para la detección, describiendo los métodos de análisis modal y métodos de identificación modal, haciendo una descripción del análisis de Prony.

En el capítulo 3, se muestran las descripciones de los programas utilizados para la realización del proyecto. En este capítulo se menciona cómo son sincronizadas las tareas; los tiempos que lleva cada tarea y la relación existente entre el programa en el microcontrolador y el método de Prony, haciendo el seguimiento de cada programa mediante distintos tipos de diagramas.

En el capítulo 4, se recopilan todas las pruebas realizadas al método de Prony partiendo de pruebas a señales ideales, pasando por una simulación a un sistema de potencia y llegando a la ejecución de la prueba final, se muestran los resultados y el análisis de cada uno de ellos.

En el capítulo 5, se enlistan las conclusiones a las que se llega con la realización de este trabajo, también se mencionan las aportaciones y se hacen algunas observaciones para realizar trabajos futuros.

CAPÍTULO 2

OSCILACIONES DE BAJA FRECUENCIA EN SISTEMAS ELÉCTRICOS DE POTENCIA

En este capítulo se realiza una breve descripción del fenómeno de las oscilaciones de baja frecuencia en los sistemas de potencia, una clasificación de los modos oscilatorios y los métodos analíticos utilizados para detectar las oscilaciones.

2.1 Introducción

En la primera parte del siglo XX, el fenómeno de las oscilaciones de baja frecuencia empezó a convertirse en un problema para los ingenieros de sistemas de potencia, en esta etapa el fenómeno de las oscilaciones de baja frecuencia fue conocido como una variación periódica de la velocidad [13]. El fenómeno hizo su aparición con la interconexión de generadores a los sistemas de potencia a través de líneas largas de transmisión.

Algunos de los problemas encontrados en esa etapa del siglo, fueron que los generadores que alimentan pequeñas cargas eran susceptibles a sufrir las llamadas variaciones periódicas, y por otro lado; cuando los generadores síncronos eran sobrecargados, tendían a perder sincronismo en una inestabilidad monótona o no oscilatoria. Estos dos fenómenos fueron definidos con el nombre de "*Estabilidad en estado estacionario"*; las oscilaciones de baja frecuencia eran debidas a un inadecuado par de amortiguamiento y la inestabilidad no oscilatoria era provocada por un inadecuado par de sincronización [13].

Después de estudios utilizando métodos analíticos para predecir la estabilidad de los sistemas de potencia, se introdujeron devanados de amortiguamiento y demostraron ser eficaces en la prevención de las oscilaciones de baja frecuencia. También se discutieron los problemas provocados por las líneas largas de transmisión en la estabilidad y en las oscilaciones y el efecto que provocarían los reguladores de voltajes en estos dos aspectos [13].

En los años 1960, el problema de las pequeñas oscilaciones en sistemas potencia presentó problemas para la operación del sistema, estas oscilaciones fueron descritas como inestabilidad dinámica, pequeñas oscilaciones u oscilaciones de baja frecuencia en sistemas de potencia [12].

Tan pronto como la confiabilidad de los SEP fue creciendo en importancia, los requerimientos para que el sistema fuera capaz de recuperarse ante fallas por la acción de un relevador se añadieron a las especificaciones del diseño del sistema. El control automático de voltaje fue utilizado para prevenir a los generadores del sistema de perder sincronismo después de una falla. Sin embargo, los sistemas de excitación de respuesta rápida, tienden a disminuir el amortiguamiento del sistema ante oscilaciones [12].

Originalmente, las oscilaciones que más afectaban eran las provocadas en generadores estrechamente unidos, para amortiguar estas oscilaciones fueron introducidos los estabilizadores de sistemas de potencia (PSS en inglés) [12].

En los años 1950 y 1960, las compañías de suministro eléctrico encontraron que podían lograr mayor confiabilidad y mejorar la economía mediante la interconexión con otras compañías, a menudo estas interconexiones eran mediante líneas de transmisión muy largas. En algunos casos, cuando las compañías realizaban estas interconexiones las oscilaciones de baja frecuencia fueron en aumento provocando que no se lograra dicha conexión. Para algunos casos el bajar la ganancia de los reguladores automáticos de voltaje fue suficiente para realizar una interconexión exitosa [12].

Desde el punto de vista de la operación del sistema, las oscilaciones son aceptables a medida que estas decaigan. Sin embargo, las oscilaciones son una característica del sistema, estas inician con pequeños cambios de carga normales en los sistemas de potencia, una operación no alarmante para el operador se considera cuando en una nueva condición de operación la oscilación decrece en magnitud [12].

Existen muchas razones para la reaparición de las oscilaciones de baja frecuencia, algunas de las más importantes son [4]:

- *Para oscilaciones entre sistemas, los devanados de amortiguamiento ya no son efectivos, ya que la amortiguación producida se reduce en proporción aproximadamente inversa al cuadrado de la eficacia de la impedancia externa más la impedancia del estator, y por lo tanto va desapareciendo.*

- *La proliferación de controles automáticos ha incrementado la probabilidad de interacciones adversas entre sistemas interconectados. incluso sin tales interacciones, los dos controles básicos del sistema como el regulador de velocidad y regulador automático de voltaje(AVR en inglés), casi siempre produce amortiguamiento negativo en el rango de frecuencias de oscilación de los sistemas de potencia, muy pequeño en los efectos del gobernador y muy grande en los AVR.*

- *A pesar de que los controles automáticos son prácticamente los únicos dispositivos que pueden producir amortiguamiento negativo, el amortiguamiento del sistema sin control en sí es muy pequeño y fácilmente podría ser provocado por los continuos cambios de carga y generación del sistema y daría como resultado oscilaciones de potencia en los lazos de interconexión.*

- *Una pequeña oscilación en cada generador, puede sumar a una oscilación en las líneas de enlace y provocar un problema significativo en relación con su clasificación.*

- *El aumento de líneas de enlace incrementa la tendencia a oscilar del sistema.*

Para calcular el efecto del amortiguamiento del sistema, el detalle de la representación del sistema tiene que ser ampliado considerablemente. El resto de parámetros requeridos para un estudio ampliado son mucho menos conocidos que las inercias del generador y las impedancias de la red necesarios para los estudios clásicos [4]. Además, el total de amortiguamiento de un sistema de potencia suele ser muy pequeño y está conformado de componentes positivos y negativos. Por lo tanto, si se desea obtener resultados reales, se deben incluir todas las fuentes conocidas. Estas fuentes incluyen: impulsores, reguladores de velocidad, cargas eléctricas, la resistencia del circuito, devanados de amortiguamiento, la excitación del generador y, de hecho, todos los controles que pueden añadirse para fines especiales. En las grandes redes, y en particular en lo que se refiere a las oscilaciones, se depende de dos elementos para producir el amortiguamiento positivo, las cargas eléctricas (al menos para generadores accionados por las turbinas de vapor) y la fuerza motriz [4].

Mayor amortiguamiento reduciría la tendencia a oscilar y la magnitud de las oscilaciones. Las razones por las cuales los sistemas de potencia a menudo son problemáticos son distintas, dependiendo de la naturaleza del sistema y las condiciones de operación.

Existe un caso específico que parece haber precipitado la idea general de brindarle mayor importancia y mejorar el amortiguamiento del sistema, así como la atención al regulador de voltaje en la generación de amortiguamiento negativo, esta fue la serie de estudios de la estabilidad transitoria de la interconexión del pacifico con la costa oeste de los Estados Unidos de América. En estos estudios, se notó que para fallas trifásicas, la inestabilidad fue determinada no por los primeros cambios en los generadores si no por la inestabilidad oscilatoria del sistema de post falla, que no tenía una de dos secciones de líneas de corriente alterna y así se incrementó la impedancia. Esto demostró que el amortiguamiento es importante para estudios transitorios como para condiciones en estado estable y ha contribuido al incremento en el uso de los estabilizadores de sistemas de potencia en los reguladores de voltaje del generador [12].

Pero el incremento de las redes eléctricas y el aumento de las cargas, han mostrado que no es suficiente solo con el uso de PSS para afrontar el problema de las oscilaciones no amortiguadas. Cuando se añade soporte de voltaje en puntos adecuados de la red, no sólo aumenta su fuerza, sino también mejora el amortiguamiento para aliviar los generadores de buena parte de la regulación de voltaje y también reducir la ganancia del regulador, sea o no el objetivo reducir el amortiguamiento [4].

Las oscilaciones en sistemas de potencia en su conjunto todavía pueden ocurrir en un sistema aislado, debido a la banda muerta del gobernador o la interacción con el sistema de control de frecuencia, pero no es probable que estos problemas ocurran entre grandes sistemas interconectados. Estas oscilaciones ocurren más en las interconexiones en una red a través de la constitución de subsistemas, especialmente a través de enlaces débiles o líneas altamente cargadas [4].

2.2 Modos de oscilación en baja frecuencia

En lo que se refiere a oscilaciones de baja frecuencia los SEP han experimentado problemas con los siguientes tipos de oscilaciones de frecuencia [11].

- *Oscilaciones de modo local.*
- *Oscilaciones de modo interárea.*
- *Oscilaciones de modo de control.*
- *Oscilaciones de modos torsionales.*

2.2.1 Oscilaciones de modo local

El modo local o modo de planta, es uno de los más comunes en los sistemas de potencia y están asociados con unidades en una estación de generación oscilando con respecto al resto del sistema de potencia. Tales problemas son usualmente provocados por la acción de los AVR de las unidades de generación operando con una salida alta y que alimentan a redes de transmisión débiles. El problema es mayor con sistemas excitación de respuesta rápida. El modo local presenta frecuencias típicas de 1 a 2 Hz [4].

2.2.2 Oscilaciones de modo interárea

Las oscilaciones interárea están asociadas con máquinas en una parte del sistema oscilando con máquinas en otra parte. Estas oscilaciones son causadas por dos o más grupos de máquinas fuertemente acopladas, que son interconectados con líneas débiles. Las frecuencias con las cuales se identifica a este modo de oscilación van de 0.1 a 1 Hz. Las características de los modos interárea de oscilación son complejos y algunos casos diferentes a las características de los modos locales de planta [4].

Los grandes sistemas interconectados existentes presentan dos formas distintas de modo interárea:

1. *Modos de frecuencia muy bajos involucran a todos los generadores en el sistema. El sistema esencialmente se divide en dos partes, con generadores de una parte oscilando contra las maquinas en la otra. Las frecuencias de estos modos de oscilación están en el orden de 0.1 a 0.3 Hz [11].*

2. *Modos de frecuencia más altos que los anteriores involucran a subgrupos de generadores contra otro, las frecuencias de estas oscilaciones se encuentran típicamente en el rango de 0.4 a 0.7 Hz [11].*

Las características de los modos interárea de oscilación son muy complejas y tienen algunas diferencias significativas con el modo local, las características de la carga, en particular, tienen mayor influencia en la estabilidad de los modos interárea.

La manera en como los sistemas de excitación afectan a los modos de oscilación interárea depende del tipo y la localización de los excitadores, y las características de la carga.

Los gobernadores de velocidad por lo regular no tienen un efecto significativo en las oscilaciones interárea. Sin embargo, si los gobernadores no son sintonizados adecuadamente, pueden provocar una ligera pérdida de amortiguamiento.

Debido a que estas oscilaciones involucran muchas máquinas, un amortiguamiento exitoso de tales modos quizá requiere la aplicación de PSS's en los sistemas de excitación de un gran número de máquinas [11].

Otros medios para lograr una estabilización efectiva de este tipo de oscilaciones incluyen la modulación de los controles de un enlace de corriente directa en alta tensión ("*HVDC*" en inglés) y los controles de un Compensador Estático de Vars (CEV) [11].

2.2.3 Oscilaciones de modo de control

Las oscilaciones del modo de control están asociadas con los controles de las unidades generadoras entre otros equipos. Sistemas de excitación mal sintonizados, primo motores, compensadores estáticos de Vars, convertidores de HVDC son las causas usuales de las inestabilidad por modo de control. En algunas ocasiones, es difícil la sintonización de controles para asegurar el amortiguamiento positivo de todos los modos [4].

2.2.4 Oscilaciones de modos torsionales

Estos modos se presentan cuando los controles de excitación, gobernador de velocidad, controles HVDC, compensadores capacitivos serie, interactúan con la dinámica del sistema eje-turbina-generador. Estos tipos de modo de oscilación presentan frecuencias de 0.7 a 2 Hz [4].

2.3 Métodos de análisis de oscilaciones de baja frecuencia

Para tener un entendimiento completo de las oscilaciones de sistemas de potencia generalmente se requiere una combinación de herramientas de análisis. Existen herramientas especializadas como el análisis modal y la identificación modal, además de las herramientas fundamentales en estudio de sistemas de potencia como flujos de carga y simulaciones de estabilidad transitoria que también son necesarias [4].

Las oscilaciones en sistemas de potencia son frecuentemente observadas en simulaciones de estabilidad transitoria realizadas en estudios de planeación u operación, también se observan oscilaciones por operadores del sistema de potencia en el seguimiento de algún disturbio o cuando es rebasado el límite de estabilidad oscilatoria. En ambos casos, las herramientas especializadas son necesarias para entender la naturaleza de las oscilaciones, para entonces llevar a cabo los procedimientos necesarios del sistema de control y así mantener la seguridad del sistema [4].

Se han desarrollado distintas técnicas para encontrar los modos que provocan las oscilaciones de baja frecuencia en el sistema, la mayoría de estas técnicas pueden ser clasificadas en dos grupos básicos [4]:

- *El análisis modal*
- *La identificación modal*

Donde el análisis modal involucra la determinación de los modos característicos del modelo linealizado del sistema acerca de un punto de operación especifico y la identificación modal involucra la determinación de modos característicos desde el comportamiento dinámico del sistema a partir de mediciones o desde simulaciones de estabilidad transitoria usando modelos no lineales [4].

La aplicación y desarrollo de las herramientas de análisis modal e identificación modal se han acelerado por la aparición de oscilaciones inestables y el bajo amortiguamiento que se han observado en la práctica o en simulaciones de planificación. La aparición de estas oscilaciones ha llevado al desarrollo de métodos especiales para permitir el análisis de modelos grandes [4].

2.3.1 Métodos de análisis modal

El análisis modal es una técnica basada en la linealización del modelo no lineal del sistema de potencia alrededor de un punto de equilibrio representado por una condición de estado estable en la operación del sistema. Una vez obtenido el modelo linealizado, se analiza el amortiguamiento de las oscilaciones con base en los modos naturales del sistema (valores propios). Si el sistema no es lo suficientemente amortiguado, las oscilaciones que se originan por cambios pequeños en la demanda, en la generación o en ajustes de controles [4] [17].

Análisis modal básico y formulación

La simulación de sistemas de potencia representa ahora un procedimiento rutinario en los sistemas de planeación y operación. Estas simulaciones utilizan ecuaciones algebraicas para representar a la red de transmisión y los dispositivos de potencia de los sistemas dinámicos, tales como los generadores y sus controles, enlaces HVDC y dispositivos FACTS. Las ecuaciones en su forma general son [4] [11] [17]:

$$\frac{dx}{dt} = f(x, v, d) \tag{2.1}$$

$$0 = g(x, v) \tag{2.2}$$

$$y = h(x, v) \tag{2.3}$$

Dónde:

x: Es un vector de variables de estado.
v : Vector de voltajes de red.
d : Vector de disturbios o entradas de control.
y : Vector de salidas monitoreadas de control o información.
f : Representa las características dinámicas no lineales de los componentes de la dinámica del sistema.
g : Representa las ecuaciones de la red no lineales.
h : Representa la ecuación de salida no lineal.

En el análisis modal, las ecuaciones anteriores son linealizadas alrededor de un punto de operación mediante las series de Taylor que expanden a f y g, para el primer término. Ya que se asume que el sistema está en estado estacionario en el punto de operación definido por x_0 y v_0, entonces $f(x_0, v_0)$ y $g(x_0, v_0)$ son cero.

Entonces las ecuaciones linealizadas son:

$$\frac{d\Delta_x}{dt} = \frac{\partial f}{\partial x}\Delta x + \frac{\partial f}{\partial v}\Delta v + \frac{\partial f}{\partial d}\Delta d \tag{2.4}$$

$$0 = \frac{\partial g}{\partial x}\Delta x + \frac{\partial g}{\partial v}\Delta v \tag{2.5}$$

$$\Delta_y = \frac{\partial h}{\partial x}\Delta x + \frac{\partial h}{\partial v}\Delta v \tag{2.6}$$

O bien,

$$\frac{d\Delta_x}{dt} = A_d\Delta x + B_{dv}\Delta v + B_{dd}\Delta d \tag{2.7}$$

$$0 = C_d\Delta x + Y_n\Delta v \tag{2.8}$$

$$0 = C_0\Delta x + K_0\Delta v \tag{2.9}$$

Las ecuaciones linealizadas, son el punto de partida de todos los programas de análisis modal de sistemas de potencia. La forma en que se utilizan depende del tamaño del modelo del sistema de potencia que se analiza, ya que incluyen las ecuaciones que rigen los dispositivos de la dinámica del sistema y ecuaciones estáticas de la red de transmisión [4].

Ecuaciones de estado del sistema

Las ecuaciones algebraicas se pueden eliminar para formar las ecuaciones de estado del sistema:

$$\frac{d\Delta_x}{dt} = \mathrm{A}\Delta x + B_{dd}\Delta d \tag{2.10}$$

$$\Delta y = \mathrm{C}\Delta x \tag{2.11}$$

Con

$$\mathrm{A} = A_d - B_{dv}(Y_n)^{-1}C_d \tag{2.12}$$

$$= C_0 - K_0(Y_n)^{-1}C_d \tag{2.13}$$

Los modos característicos de la ecuación 2.12 tiene la forma general $u_i e^{\lambda_i t}$. El número de modos característicos es igual al número de variables de estado en el modelo. El vector u_i es llamado vector característico (eigenvector) y el valor de λ_i es nombrado valor característico (eigenvalor). Los eigenvalores reales indican que son aperiódicos.
Los eigenvalores complejos indican modos que son oscilatorios, para eigenvalores complejos de la forma $\sigma \pm j\omega$, la amplitud del modo variará de la forma $e^{\sigma t}$ y la frecuencia de oscilación será $\omega/2\pi$. La relación de amortiguamiento ζ está definida como:

$$\zeta = \frac{-\sigma}{\sqrt{\sigma^2 + \omega^2}} \tag{2.14}$$

Los eigenvalores satisfacen la ecuación:

$$det(A - \lambda_i I) = 0 \tag{2.15}$$

Los eigenvectores derechos son vectores columna que satisfacen:

$$Au_i = \lambda_i u_i \tag{2.16}$$

Los eigenvectores izquierdos son vectores fila que satisfacen:

$$v_i \mathrm{A} = \lambda_i v_i \tag{2.17}$$

Eigenvalores

Para que el punto de operación del sistema sea considerado estable, la parte real de los eigenvalores de la matriz A debe ser negativa. Esto implica que después de un disturbio pequeño, todos los modos decaerán con el tiempo y el sistema regresará a un estado estable. En el caso de que alguno de los eigenvalores tenga parte real positiva, entonces después de un pequeño disturbio la amplitud del modo crecerá exponencialmente con el tiempo de tal manera que domina la dinámica del sistema, en tal caso, el sistema de potencia es inestable en ese punto de equilibrio [4][17].

Eigenvectores

Los eigenvectores derechos están asociados con cada modo y definen la distribución relativa del modo a través de los estados de la dinámica del sistema. Esta característica se utiliza cuando se comparan las simulaciones de estabilidad transitoria con los resultados obtenidos del análisis modal. Si el resultado de la simulación está dominado por un solo modo, la relación entra la amplitud de cualquier estado con cualquier otro en la simulación corresponden a la relación de las magnitudes de los vectores propios asociados a dicho modo [4].

Los eigenvectores no son únicos, ya que dependen de la elección de los estados realizada en la formulación del modelo. Los eigenvectores no son adimensionales, esto significa que los eigenvectores no necesariamente indican la importancia de un estado particular en un modo [4].

Los eigenvectores izquierdos se pueden interpretar como una distribución de los estados de un modo, esto tiene un efecto directo sobre la amplitud de un modo excitado por una entrada específica [4].

2.3.2 Métodos de identificación modal

En un sistema actual, es posible que después de un disturbio, se pueda registrar el comportamiento transitorio del sistema. Para extraer los diferentes modos de oscilación de los registros transitorios, es necesario utilizar herramientas especiales de identificación modal, esto es particularmente útil en un entorno operativo [4].

Debido a que las herramientas de identificación modal se pueden utilizar con las respuestas ya sean simuladas o medidas, son muy útiles para la validación de modelos de simulación lineales y no lineales contra del comportamiento observado en el sistema [4].

La identificación modal es una técnica más experimental que el análisis modal, por lo que no existe una guía estándar que categorice los casos en los cuales las técnicas de identificación modal presentan mejores resultados [4].

Las técnicas de identificación modal también se prestan a la determinación de modelos estructurados de sistemas de potencia y dispositivos de control. Esta cualidad es a menudo necesaria: si el dispositivo tiene no linealidades inherentes o características que no pueden ser completamente resueltas por el análisis modal.

Existen muchas técnicas de análisis de señales como: la correlación en el tiempo, el análisis del espectro paramétrico y la transformada de ondeletas. También existen herramientas básicas como el análisis de Fourier y el método de Prony [4].

2.3.2.1 El Método de Prony

2.3.2.1.1 Introducción al análisis de Prony

Gaspard Richie, Baron de Prony en 1795, al analizar los efectos de la presión del vapor del alcohol, observó que las combinaciones lineales de exponenciales complejas obedecían a recursiones lineales homogéneas y que estas podían usarse para interpolar o ajustar datos [19].

El análisis de Prony tuvo que esperar la aparición de la computadora digital y medios adecuados para tratar con problemas matemáticos inherentemente mal condicionados [1].

Fundamentalmente, el análisis de Prony es un método de ajuste de una combinación lineal de términos exponenciales a un número finito de muestras de una señal igualmente espaciada en el tiempo. Las ventajas numéricas del análisis de Prony lo hacen adecuado para la aproximación de las señales de alto orden con un óptimo orden inferior del modelo [5].

El método de Prony asume que el sistema es de una sola salida en contraste con la realidad de los SEP´s que son de salida múltiple (muchas señales que oscilan con el mismo modo) [5]. El método analiza señales individuales de forma independiente, lo que a menudo resulta en estimaciones de modo conflictivas. La experiencia tomada por el operador del método, es quien tiene la problemática de determinar que estimaciones modales son las más precisas, en muchos casos esta resulta una tarea imposible. La señal del ruido es la principal limitante en la obtención de estimaciones precisas de los modos en el análisis de Prony [5].

El análisis de Prony se relaciona con el de Fourier ya que los dos se pueden expresar como suma de exponenciales. El método de Prony es capaz de estimar directamente la frecuencia, también es posible conocer el amortiguamiento, la magnitud y la fase relativa de las componentes modales identificadas en una señal dada.

2.3.2.1.2 Descripción del método de Prony

Un sistema dinámico invariante en el tiempo y lineal con valores iniciales $x(t_0) = x_0$ en el tiempo t_0, se comportará de acuerdo a la siguiente ecuación diferencial [10]:

$$\dot{x} = Ax \tag{2.18}$$

La cual tiene una solución en términos de sus matrices residuales:

$$\dot{x}(t) = \sum_{i=1}^{n} R_i x_0 e^{\lambda_i t} \tag{2.19}$$

Para N muestras complejas, el modelo exponencial propuesto es:

$$\hat{x}(n) = \sum_{k=1}^{p} A_k \, e^{[(\alpha_k + j2\pi f_k)nT + j\theta_k]}; \quad 0 \le n \le N-1 \tag{2.20}$$

O también:

$$\hat{x}(n) = \sum_{k=1}^{p} h_k \, z_k^n; \;\; 0 \le n \le N-1 \tag{2.21}$$

Donde:

$$h_k = A_k e^{j\theta_k} (amplitud\ compleja) \tag{2.22}$$

$$z_k = e^{(\alpha_k + j2\pi f_k)T} (exponencial\ compleja) \tag{2.23}$$

Dónde: p es el número de exponenciales complejas, T es el periodo de muestreo en segundos, A_k es la amplitud de la exponencial compleja, a_k es el factor de amortiguamiento en segundos f_k es la frecuencia en Hz, y θ_k es la fase en radianes.

De acuerdo con la ecuación (2.21) existen $2p$ parámetros desconocidos, es decir, $h1, h2, .. hp, z1, z2, .. zp$. O sea que se requieren $N = 2p$ muestras $x(1), \ldots . x(2p)$ para obtener las $2p$ incógnitas. La ecuación (3.4) puede expresarse en la siguiente forma matricial para $0 \le n \le p-1$.

$$\begin{bmatrix} x(0) \\ x(1) \\ \vdots \\ x(N-1) \end{bmatrix} = \begin{bmatrix} z_1^0 & z_2^0 & \ldots & z_p^0 \\ z_1^1 & z_2^1 & \ldots & z_p^1 \\ \vdots & \vdots & \ddots & \vdots \\ z_1^{N-1} & z_2^{N-1} & \ldots & z_p^{N-1} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_p \end{bmatrix} \tag{2.24}$$

La contribución de Prony consistió en descubrir la forma de desacoplar h_k y z_k utilizando las p muestras restantes. La clave está en demostrar que la ecuación 2.25 es la solución de una ecuación diferencia $\varphi(z)$ de coeficientes constantes (ecuación 2.26).

$$\hat{x}(n) = \sum_{k=1}^{p} h_k \, z_k^n; \;\; p \leq n \leq p-1 \tag{2.25}$$

$$\varphi(z) = \prod_{k=1}^{p}(z - z_k) = z^p + a(1)z^{p-1} + a(2)\, z^{p-2} + \cdots + a(p) = 0 \tag{2.26}$$

Si los productos de ecuación (2.26) se expanden en una serie de potencias, entonces el polinomio puede representarse como un polinomio de la forma:

$$\varphi(z) = \sum_{m=0}^{p} a(m) z^{p-m} = 0 \qquad (a(0) = 1) \tag{2.27}$$

Multiplicando la ecuación (2.25) por $a(m)$, reemplazando n por $n - m$ y sumando para $m = 0, 1, 2, .. p$, se tiene:

$$\sum_{m=0}^{p} a(m)x(n-m) = \sum_{m=0}^{p} a(m) \sum_{k=1}^{p} h_k \, z_k^n \tag{2.28}$$
$$p \leq n \leq 2p-1$$

Sustituyendo la ecuación (2.29) en la ecuación (2.28).

$$z_k^{n-m} = z_k^{n-p} z_k^{p-m} \tag{2.29}$$

$$\sum_{m=0}^{p} a(m)x(n-m) = \sum_{m=0}^{p} a(m) z_k^{p-m} \sum_{k=1}^{p} h_k z_k^{n-p} = 0 \tag{2.30}$$

De la ecuación (2.27) $\sum_{m=0}^{p} a(m) z_k^{p-m}$=0 y desarrollando la parte izquierda de la ecuación (2.30) obtenemos:

$$x(n) + \sum_{m=1}^{p} a(m)x(n-m) = 0 \tag{2.31}$$

Entonces despejando de la ecuación (2.31):

$$x(n) = -\sum_{m=1}^{p} a(m)x(n-m) \quad p \leq n \leq 2p-1 \tag{2.32}$$

La ecuación (2.32) es la ecuación diferencia lineal cuya solución homogénea está dada por la ecuación (2.21) y puede expresarse en la siguiente forma matricial:

$$\begin{bmatrix} \mathrm{x(p)} & \mathrm{x(p-1)} & \dots & \mathrm{x(1)} \\ \mathrm{x(p+1)} & \mathrm{x(p)} & \dots & \mathrm{x(2)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{x(2p-1)} & \mathrm{x(2p-2)} & \dots & \mathrm{x(p)} \end{bmatrix} \begin{bmatrix} \mathrm{a(1)} \\ \mathrm{a(2)} \\ \vdots \\ \mathrm{a(p)} \end{bmatrix} = \begin{bmatrix} \mathrm{x(p+1)} \\ \mathrm{x(p+2)} \\ \vdots \\ \mathrm{x(2p)} \end{bmatrix} \tag{2.33}$$

De esta ecuación se demuestra que con 2p muestras es posible desacoplar los parámetros h_k y z_k.

Cálculo de la frecuencia de oscilación y factor de amortiguamiento.

Se calculan los eigenvalores a partir de las raíces del polinomio de la ecuación 2.26.

$$\lambda = \frac{1}{T} ln(z_k) \tag{2.34}$$

$$a_k = real(\lambda)\, seg^{-1} \;\; (factor\ de\ amortiguamiento) \tag{2.35}$$

$$f_k = \frac{\omega}{2\pi} \;\; Hz \quad (frecuencia\ de\ oscilacion) \tag{2.36}$$

Cálculo de la relación señal-ruido

La señal reconstruida $\hat{x}$ usualmente ajustará a $x(t)$ inexactamente. El parámetro que describe la calidad del ajuste de señales es denominado *SNR* ("*signal to noise ratio*", por sus siglas en inglés), y está definido como en [3]:

$$SNR = 20 \log \; rms \left(\frac{x(n)}{x(n) - \hat{x}(n)} \right) \dots . db \tag{2.37}$$

Una buena precisión en la aplicación del Análisis de Prony es lograda para valores del SNR alrededor de 40 db; valores más bajos del SNR pueden dar lugar a soluciones erróneas del método de Prony y por lo general implica que el orden de predicción del modelo lineal es demasiado bajo [3].

2.3.2.1.3 Algoritmo de Prony

El proceso para la obtención de las incógnitas requiere de tres pasos básicos adaptados de [2][3][5]:

PASO 1.- Determinar los parámetros de predicción lineal que se ajustan a los datos disponibles (formar y resolver ecuación 2.33).

PASO 2.- Encontrar las raíces del polinomio de predicción del paso 1, encontrar los coeficientes de predicción que producirán las estimaciones de factor de amortiguamiento y frecuencia sinusoidal de cada termino exponencial (solución ecuación 2.26 y 2.34).

PASO 3.- Con las raíces encontradas en el paso 2, obtenemos una segunda ecuación lineal, con la cual es posible estimar la amplitud de la exponencial y la fase inicial de la sinusoidal (formar y resolver ecuación 2.24).

En base a este procedimiento se diseña un algoritmo a seguir para el método de Prony que se muestra en la figura 2.2

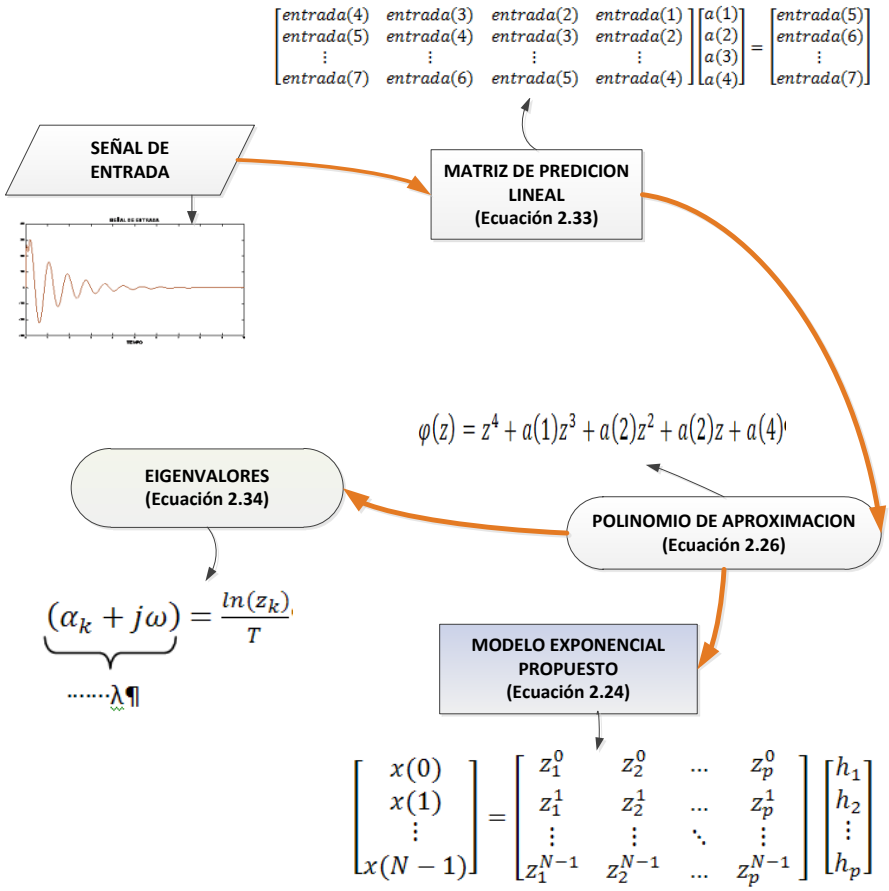


Figura 2.1. Algoritmo general de Prony.

2.3.2.1.4 Observaciones del método de Prony

De manera simultánea a los modos identificados con el análisis de Prony, se encuentran los llamados modos accesorios que representan al ruido inherente en la señal de entrada [3].

El orden del modelo de predicción lineal debe ser mayor al orden de la señal de entrada analizada ya que los modos accesorios son necesarios para aumentar el SNR [3].

En la práctica, el análisis de Prony se aplica a sistema no lineales, se debe tener cuidado en la interpretación de los resultados del ajuste. Si la comparación directa con el análisis modal es necesaria, la señal debe ser analizada cerca del final del disturbio y no en los principios donde la amplitud del transitorio puede ser grande [5].

En el análisis de Prony, como en otros procedimientos de análisis de señales, normalmente necesitan una revisión y procesamiento previo de la señal que se desea analizar, además de un cierto grado de experiencia [4]. Algunas preguntas generales que deben ser contestadas son:

1.- ¿Cuánto de los registros debe ser procesado?
2.- ¿Cuánto de las tendencias o compensaciones debe ser removido?
3.- La forma en cómo detectar no linealidades y entradas ocultas.
4.- ¿Cómo mitigar los efectos del ruido, no linealidades y entradas ocultas?
5.- ¿Cómo determinar el orden de ajuste del modelo?

Con el análisis de Prony, el modelo ajustado es más inmediatamente identificable en términos de funciones de transferencia y constantes de tiempo. Si los modelos de la dinámica subyacente son requeridos, el uso coordinado de diferentes recursos y herramientas de análisis son normalmente necesarias. El uso complementario de una variedad de herramientas de análisis es el enfoque más beneficioso para el análisis de sistemas de potencia y la identificación de modos en el mismo [4].

CAPÍTULO 3

DISEÑO DE LA PROGRAMACIÓN

En este capítulo se expone detalladamente los programas desarrollados para el proyecto. Describiendo dos tipos de programación que se relacionan a través de una interfaz de comunicación (RS-232), se utilizó para la parte de medición y cálculo de variables la programación en tiempo real en el programa CodeWarrior 7.2 para microcontroladores de la familia Coldfire V2 utilizando como plataforma MQX 3.6 [20][21]; para el algoritmo de Prony y la interfaz de comunicación se utilizó el programa MATLAB.

Para explicar el software utilizado, se hace una descripción general del proyecto definiendo así los requerimientos bajo los cuales se realizó el diseño final; una vez explicado el concepto general del proyecto, se definen las partes que lo conforman en forma particular.

Para el desarrollo de la primer parte del proyecto, se utiliza un microcontrolador que realiza la medición de fasores, cálculo de variables y actúa como servidor al enviar los valores calculados. Después, el programa en MATLAB es el cliente al recibir las variables e introducirlas como señal de entrada al método de Prony, quien finalmente realiza la exposición de los resultados en MATLAB.

3.1 Requerimientos generales para el software implementado

Se realizó una implementación utilizando un sistema de potencia de dos áreas cargado en un simulador digital (OPAL-RT), de donde se obtienen fasores la medición de variables como potencia activa, potencia reactiva y magnitud de voltaje, estos cálculos se obtienen a partir de los fasores de las señales medidas dentro del microcontrolador; una vez realizadas las mediciones y cálculos se envían a un programa de detección con el algoritmo de Prony programado (este envió se ejecuta mediante la interfaz de comunicación RS-232), En el programa de Prony se realiza un análisis de oscilaciones de una señal de entrada con una ventana de muestra de 10 segundos, el tiempo de muestreo de la señal de entrada es de 0.03 segundos.

Como parte adicional a las prueba en línea, se utiliza una memoria USB que almacena las variables calculadas con el microcontrolador; este almacenamiento en memoria también es programado en tiempo real, esto se realiza para obtener un respaldo de información y permitir realizar pruebas posteriores y análisis más detallados. Este procedimiento se muestra de manera conceptual en la figura 3.1.

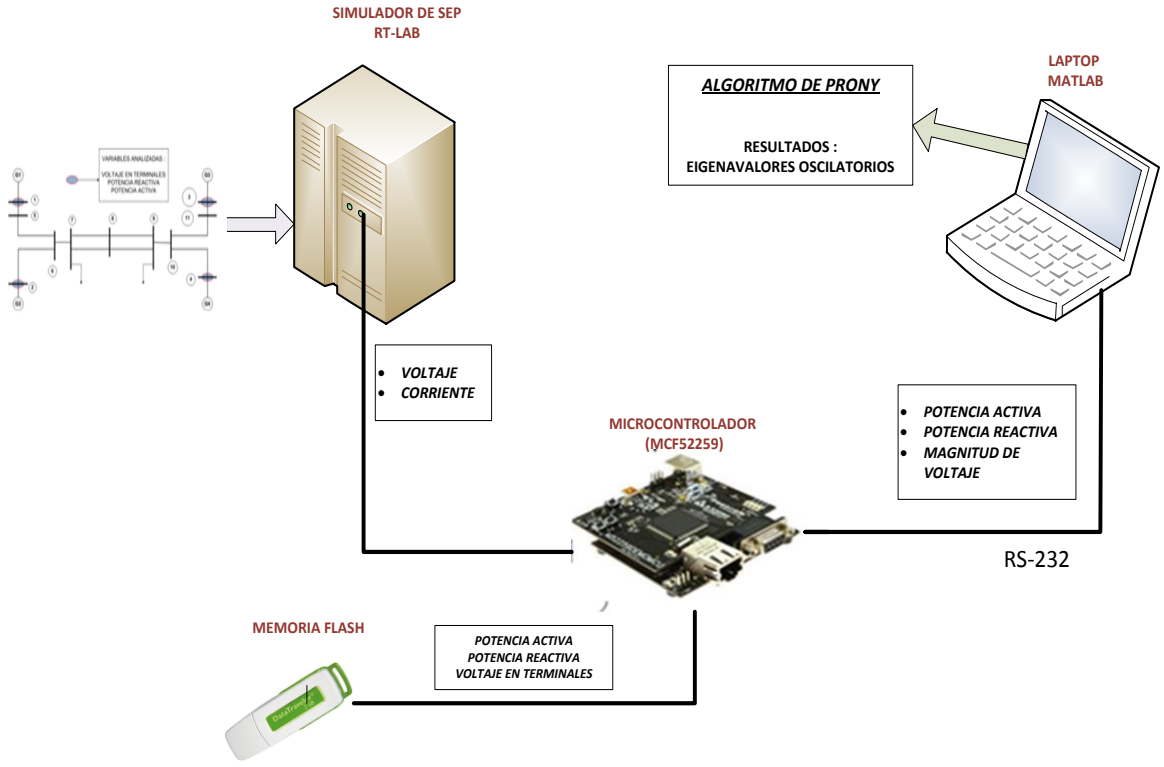


Figura 3.1. Esquema general de proyecto implementado.

Como se puede observar en la figura 3.1, se utilizan dos programas principales que se enlazan a través del cable RS-232; los dos programas trabajan de manera independiente y la comunicación entre ellos se efectúa cada 0.03 segundos (tiempo que tarda el microcontrolador en obtener el cálculo de una variable). En la figura 4.2 se muestra la interacción entre los dos programas elaborados.

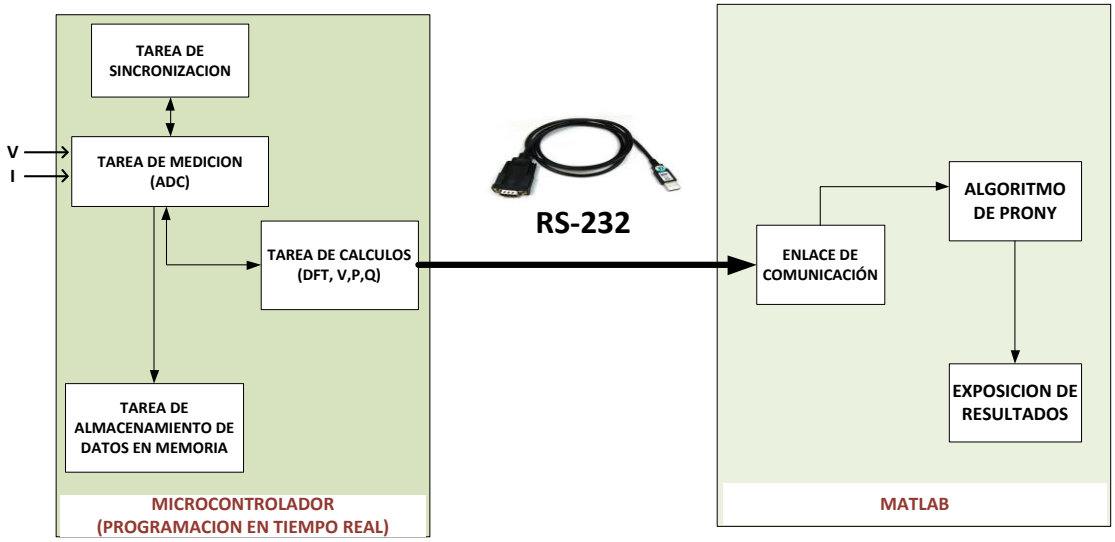


Figura 3.2. Partes generales del proyecto desde el punto de vista del software.

3.2 Programación del microcontrolador

Para la programación del Microcontrolador MCF52259, se utilizó un ambiente de desarrollo integral llamado "CodeWarrior" de Metroworks, utilizando el sistema operativo MQX 3.6 como plataforma para la realización de la programación en tiempo real.

La programación en tiempo real utilizada en sistemas embebidos presenta algunas características favorables para ciertos requerimientos tales como: uso de menos espacio de memoria en su ejecución, cualquier evento en el soporte físico puede hacer que se ejecute alguna tarea y por ultimo presenta una flexibilidad en cuanto a arquitecturas ya que puede ejecutarse en otro microprocesador [22].

En este tipo de programación, se trabaja a partir tareas que pueden presentarse en tres estados: en ejecución, prepara en espera (suspendida) y bloqueada. Otro aspecto importante, es que solo es posible ejecutar una tarea a la vez. Las tareas son activadas o desactivadas utilizando comandos de bloqueo o retardos de tiempo programados, también son ejecutadas en orden de prioridad a través de una lista de tareas preparadas.

Para realizar un proyecto en un sistema operativo en tiempo real, es importante tomar algunas consideraciones previas; entre las cuales se pueden mencionar: definir los requerimientos para el proyecto; las tareas necesarias para cumplir con dichos requerimientos; asignarle prioridad a las tareas y la secuencia de sincronización entre ellas.

Los requerimientos para nuestro proyecto son: realizar mediciones de ciclos completos de voltaje y corriente de manera simultánea y a partir de estas, obtener los fasores; de los cuales se realizan cálculos de potencia activa y reactiva que se envían al programa de Prony. Se requiere calcular 30 fasores por segundo sincronizados mediante un pulso por segundo interno.

Para dichos requerimientos, se diseña un programa que trabaja utilizando 4 tareas: Sincroniza, ADC, USB y Calcula, enumeradas de mayor a menor prioridad en nuestro proyecto.

En cuanto a la secuencia de sincronización, como se puede observar en la figura 3.3, el programa comienza realizando inicializaciones indispensables para utilizar el convertidor analógico digital (ADC en inglés) y la memoria USB. Después se empieza el ciclo de mediciones siendo el comienzo y fin de la secuencia la tarea “Sincroniza”; la tarea “ADC” es quien coordina el cálculo de las variables que se realizan en la tarea “Calcula” en donde se encuentran todos los algoritmos de medición utilizados y envía los datos al algoritmo de Prony.

Este ciclo de mediciones es repetido según sea el número de variables que se necesiten guardar, para este caso se utiliza un rango de medición de 300 valores por variable. Las variables son guardas creando un archivo de texto por cada variable.

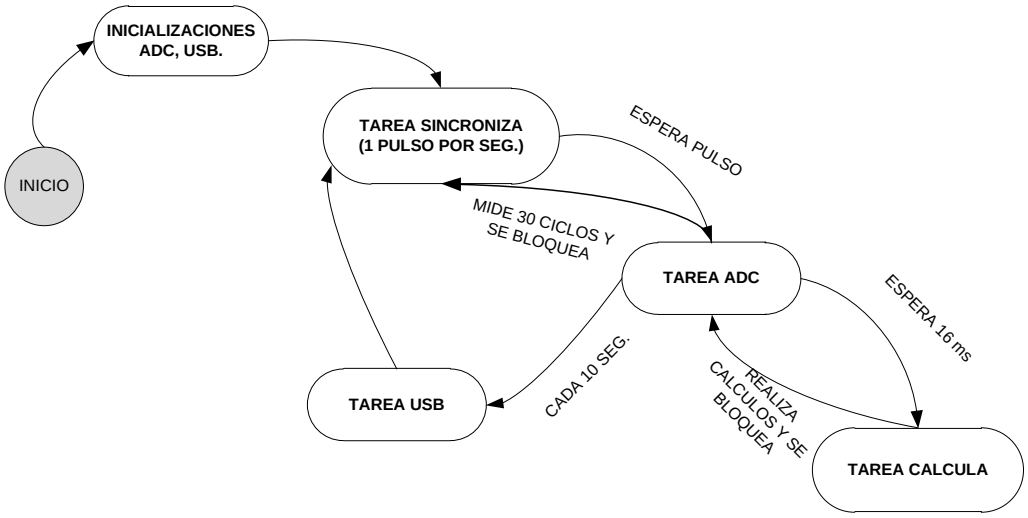


Figura 3.3. Sincronización de las tareas.

Una vez realizadas las inicializaciones, el ciclo general del programa se lleva acabo ejecutando primero la tarea “Sincroniza” que espera un pulso por segundo. Al recibir el pulso se desbloquea la tarea “ADC” y la tarea “Sincroniza” se coloca en modo de espera.

En la tarea “ADC” se obtienen una serie de mediciones; provenientes de los dos canales programados de medición correspondiente uno al voltaje y el otro a la corriente, que obtienen 64 muestras para un ciclo completo de mediciones, una vez obtenidas se activa a la tarea “Calcula” para ejecutar los algoritmos de medición; mientras estos se realizan, la tarea “ADC” se coloca en modo de espera por un ciclo completo en 60 Hz (16.66 ms), está actividad se repite 30 veces en un segundo; una vez realizadas las 30 mediciones las tareas se bloquean y esperan por el siguiente pulso de sincronización.

El número de mediciones totales a guardar en la memoria, está regido por la capacidad de almacenamiento disponible. Para el caso de las pruebas, se utilizó la medición de 300 fasores obtenidos en 10 segundos de repetición del ciclo expuesto anteriormente; una vez pasados estos 10 segundos, la tarea “USB” realiza el almacenamiento de las mediciones en la memoria física, una vez guardadas las 300 mediciones por variable es posible regresar a la repetición de las tareas de medición y cálculo.

Para ilustrar el procedimiento descrito anteriormente, en la figura 3.4 se puede observar en forma de diagrama de flujo la secuencia que sigue cada tarea y la relación que tienen con las otras.

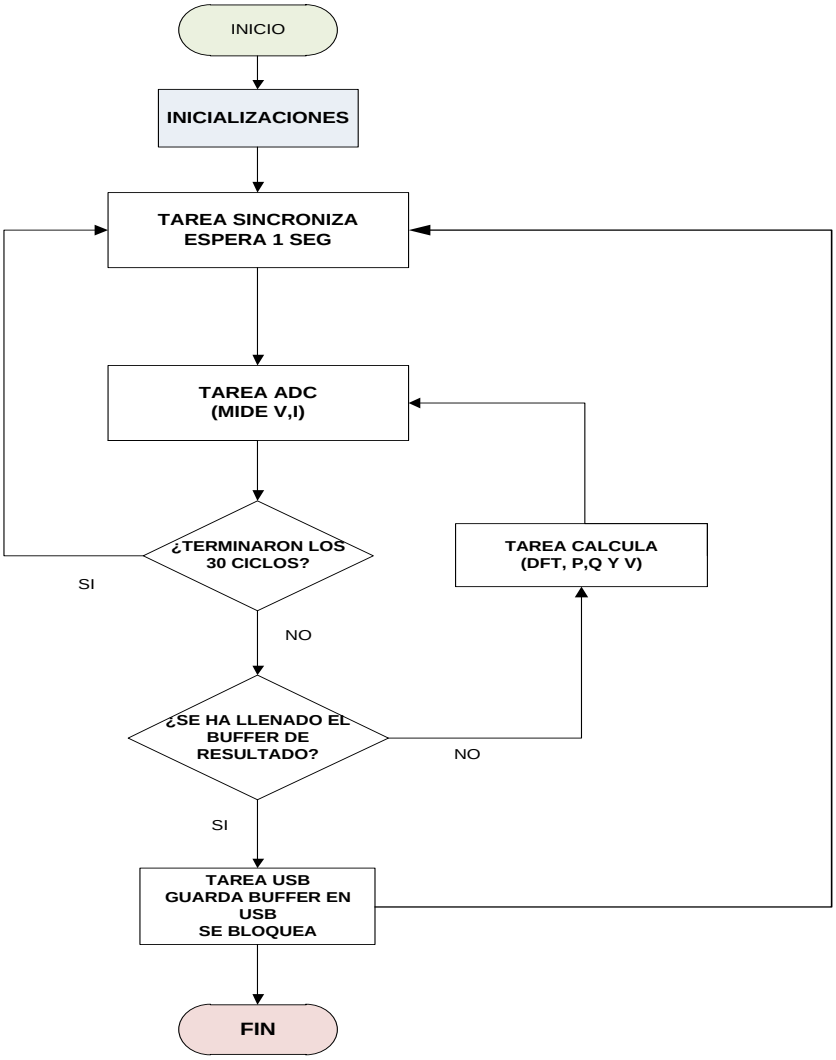


Figura 3.4. Diagrama de flujo de programa de medición de variables.

Para la obtención de buenos resultados en la programación en tiempo real es necesario un control estricto del tiempo de ejecución de cada tarea, para el caso particular del proyecto realizado se siguió una estrategia de bloqueo y desbloqueo de tareas, así como poner tareas en espera mediante retardos de tiempo.

La distribución del tiempo en las tareas “ADC”, “Calcula” y “Sincroniza” es importante, ya que un retardo no anticipado en cualquiera de ellas llevaría a obtener cálculos o mediciones erradas de la señales de entrada y salida del programa. En la figura 3.5 se muestra una gráfica mostrando la secuencia del programa poniendo énfasis en el tiempo en el cual se ejecuta cada tarea.

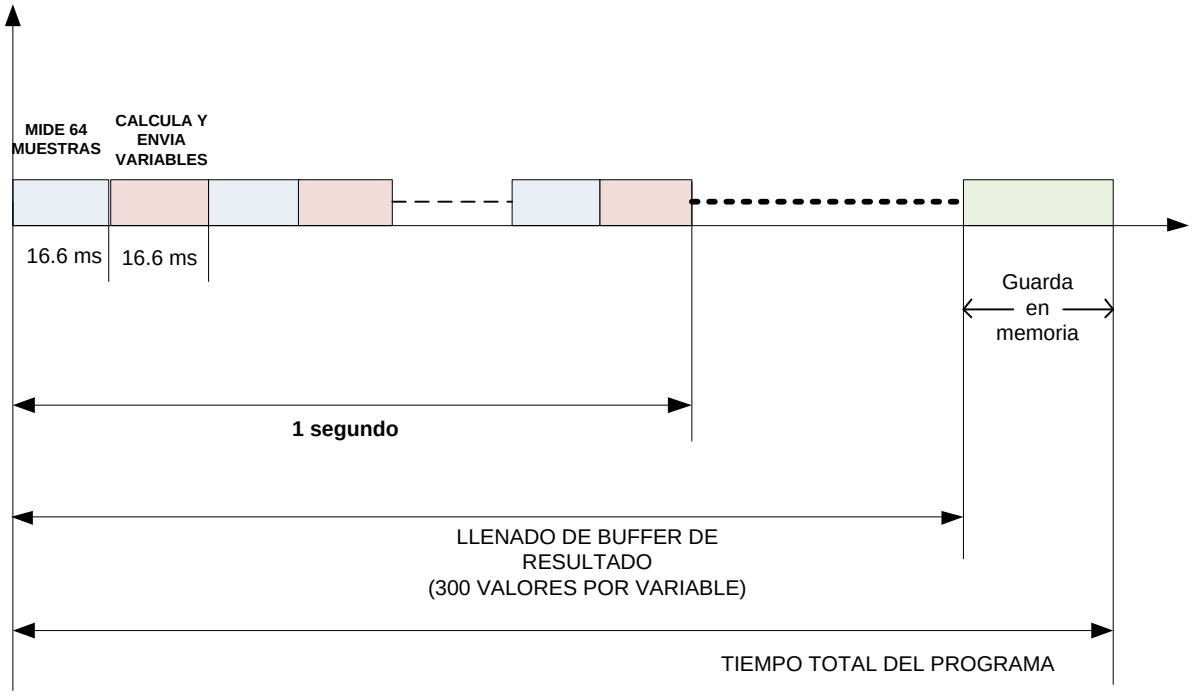


Figura 3.5. Diagrama de secuencia de tareas.

3.2.1 Tarea Sincroniza

La tarea “Sincroniza” en el proyecto es la encargada de entregar el pulso por segundo que simula la función que realiza un GPS en un unidad medidora fasorial (PMU en inglés), el pulso de sincronización marca el inicio de la medición de cada segundo, que permite que todos los medidores instalados en un sistema de potencia arranquen sus mediciones al mismo tiempo dando como resultado mediciones sincronizadas de fasores.

La tarea “Sincroniza” comienza realizando la inicialización para los leds que marcan en forma física el pulso por segundo en el microcontrolador, después hace un bloqueo temporal para permitir a las demás tareas ejecutar sus inicializaciones; una vez obtenido el desbloqueo, la tarea “Sincroniza” entra en el ciclo infinito en donde se programa una espera de un segundo, tiempo durante el cual la tarea se encuentra suspendida. En la figura 3.6 se expone el procedimiento mediante un diagrama de estados de la tarea especificada; En esta figura se marca con un círculo el periodo de trabajo de la tarea.

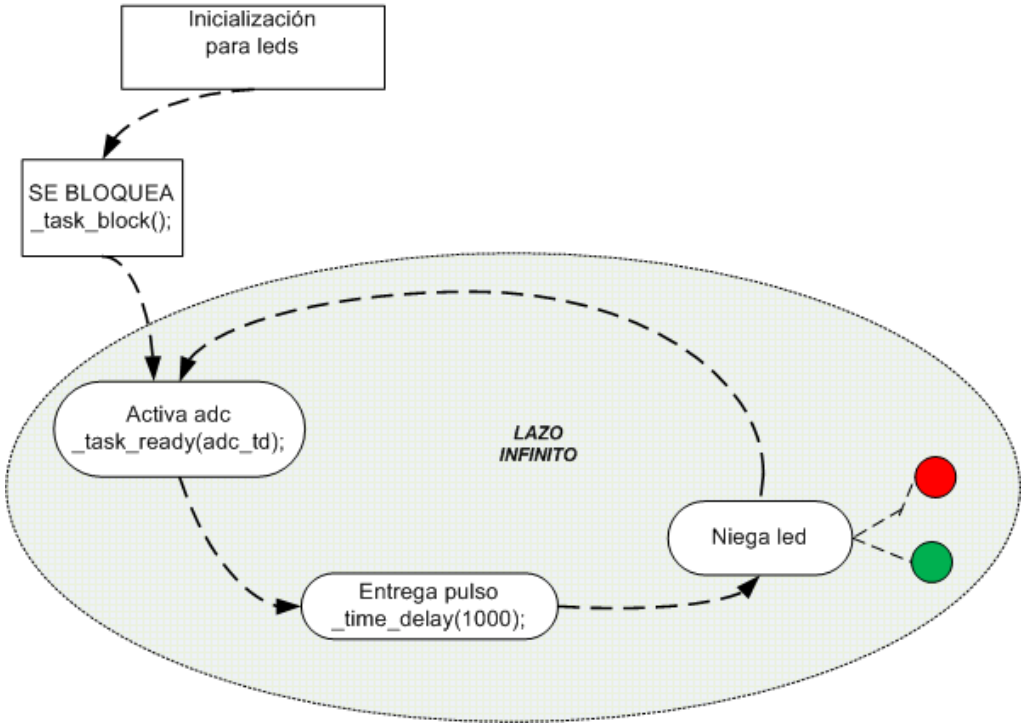


Figura 3.6. Diagrama de estados tarea "Sincroniza".

3.2.2 Tarea ADC

Esta tarea debe su nombre al convertidor analógico digital (ADC en inglés) que se encarga de entregar al programa las mediciones digitales de las señales medidas, también realiza la coordinación de las tareas "Calcula" y "USB".

La programación del ADC en tiempo real se ejecuta a través de una estructura en donde se especifican las características de funcionamiento del ADC. El microcontrolador utilizado consta de un ADC de 12 bits y utilizando dos de sus ocho canales (AN0 y AN2) programados para obtener 64 muestras por ciclo, con un periodo de muestreo de 260 µs, en esta estructura se especifica también un vector en el cual se guardan las 64 muestras, el ADC pone activo un evento el cual se utiliza en parte del código como una indicación de que el ADC ha concluido sus mediciones.

En la tabla 3.1 se enumeran las características requeridas por nuestro proyecto para realizar la medición de la señal de entrada de uno de los canales.

Tabla 3.1. Características de programación de uno de los canales del ADC utilizado.

const ADC_INIT_CHANNEL_STRUCT vL1_channel_param =	
Canal físico del ADC (0x00)	ADC_SOURCE_AN0
Medición continua después de realizado el disparo	(ADC_CHANNEL_MEASURE_LOOP | ADC_CHANNEL_START_TRIGGERED)
Número de muestras en una secuencia(64)	SAMPLES_PER_PERIOD
Tiempo de compensación utilizado por el disparo (0)	ADC_VOLTAGE_DELAY
Tiempo de muestreo (260)	ADC_SAMPLING_PERIOD
Rango de escala(no usado)	ADC_RESULT_RANGE_SCALE
Tamaño del buffer circular (64)	SAMPLE_BUFFER
Identificador del disparo del ADC (0x0001)	ADC_TRIGGER_1
Variable del evento ligero	&meter_shared_event
Mascara para el evento ligero (1)	ADC_DATA_READY_MASK

La tarea “ADC” arranca con inicializaciones indispensables tanto para el ADC en general, como de los canales a utilizar; a cada inicialización se le asigna un archivo de referencia que se utiliza para hacer el disparo del ADC y usar las mediciones, una vez terminadas; se realiza un bloqueo para permitir a la tarea “USB” terminar las inicializaciones, este bloqueo es desactivado una vez que la tarea “Sincroniza” indica que el pulso de sincronización ha sido ejecutado. Obtenido el desbloqueo, la tarea entra en un ciclo infinito el cual comienza con la medición de un ciclo completo, obtenida la medición la tarea se suspende mediante un retardo de tiempo de un ciclo completo, activando a la tarea “Calcula” para que realice los cálculos necesarios en el proyecto.

Este proceso se repite durante un segundo, a cada segundo la tarea espera por el nuevo pulso de sincronización, el ciclo se repite hasta que la medición llega a 300 valores. El procedimiento se observa en la figura 3.7 mediante un diagrama de estados.

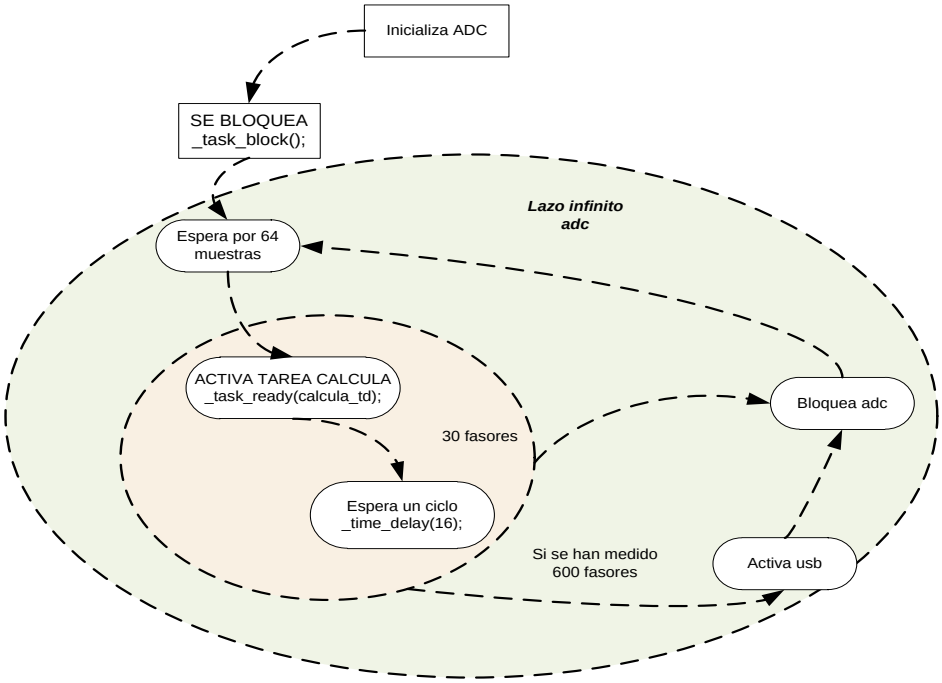


Figura 3.7. Diagrama de estados de la tarea "ADC".

3.2.3 Tarea Calcula

La tarea "Calcula" utiliza el algoritmo de la transformada discreta de Fourier para encontrar los fasores correspondientes a cada una de las variables medidas (voltaje y corriente) una vez obtenidos dichos fasores mediante fórmulas matemáticas, se estima la potencia activa y la potencia reactiva, esta tarea es activada por la tarea "ADC" y bloqueada por si misma al terminar los cálculos.

El algoritmo utilizado para la obtención de fasores es la DFT, de la cual se realiza una descripción en el siguiente apartado.

3.2.3.1 Transformada discreta de Fourier

La transformada discreta de Fourier (DFT en inglés) opera con una señal muestreada, a partir de esta 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 [16].

El análisis de la 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 [18].

Si se cuenta con la señal en el dominio del tiempo, el proceso para conocer el dominio de la frecuencia es llamado descomposición o transformada discreta de Fourier [18].

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 para nuestro caso 64, esto es debido a que los datos guardados digitalmente en una computadora usan direccionamiento binario [23].

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 [18].Estas funciones son generadas de las ecuaciones siguientes [16]:

$$ReX = \frac{\sum_{i=0}^{N-1} muestra(i) * cos(i)}{\frac{N}{2}} \tag{3.1}$$

$$ImX = \frac{\sum_{i=0}^{N-1} muestra(i) * sen(i)}{\frac{N}{2}} \tag{3.2}$$

$$magnitud = \sqrt{(Im\,X)^2 + (Re\,X)^2} \tag{3.3}$$

$$angulo = \tan^{-1}\frac{Im\,X}{Real\,X} \tag{3.4}$$

Donde:

$muestra$ = Representa cada muestra obtenida de la señal muestreada.
X = 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 $N - 1$.
$Im\,X$ y $Re\,X.$ = Son las componentes rectangulares del vector correspondiente a la descomposición de la señal analógica.

En la figura 3.8 se muestra ejemplo de la utilización transformada de Fourier con una señal de voltaje de 127 Volts a 60 Hz para la cual se calcularon sus componentes rectangulares, la magnitud y el ángulo.

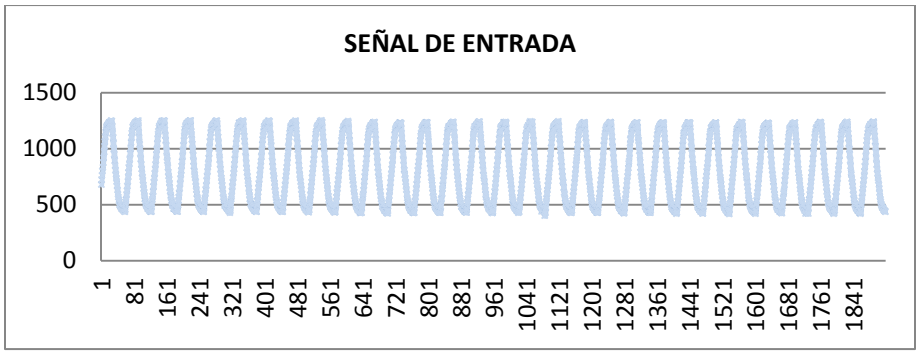


Figura 3.8. Muestra de una señal de entrada medida por el ADC.

Con esta señal de voltaje se descompone mediante la DFT en estas señales de parte real y parte imaginaria defasadas 90°.

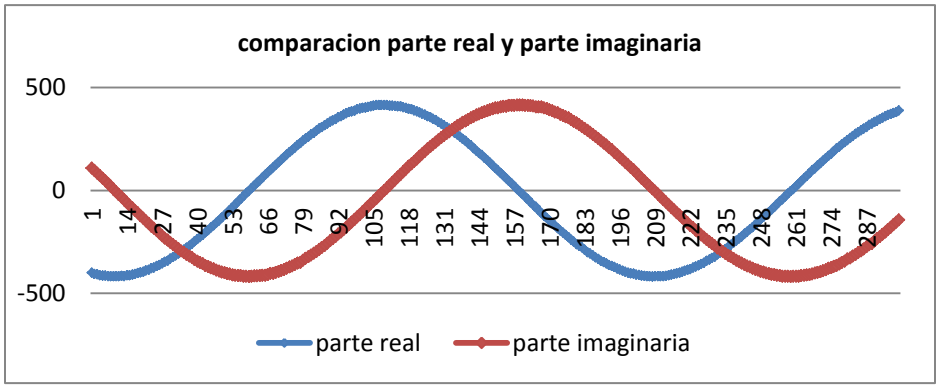


Figura 3.9. Componentes rectangulares de la señal de entrada.

Una vez obtenidas la parte real y parte imaginaria de una señal se calculan mediante la ecuación 3.3 y 3.4 la magnitud y el ángulo del fasor respectivamente, para la prueba realizada se muestran estas características en las figuras 3.10 y 3.11.

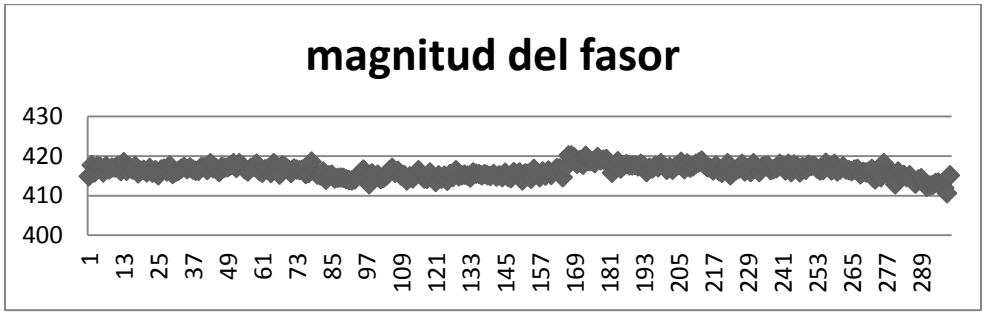


Figura 3.10. Magnitud de la señal de entrada.

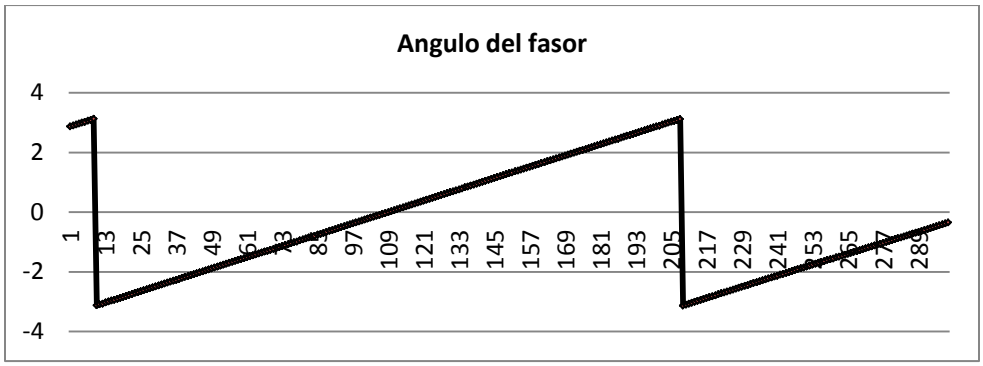


Figura 3.11. Ángulo de la señal de entrada.

A partir de las mediciones mostradas, es posible obtener otros variables de interés para el diagnóstico de un sistema de potencia. Variables como potencia activa y potencia reactiva entre otras.

3.2.3.2 Cálculo de potencia activa y potencia reactiva

Se necesita realizar el cálculo de potencia activa y potencia reactiva a partir de las componentes rectangulares de las señales de voltaje y corriente calculadas previamente utilizando la transformada discreta de Fourier.

El cálculo de la potencia activa y reactiva se realizó utilizando las ecuaciones 3.5 y 3.6:

$$P = (Re(V) * Re(I)) + (Im(V) * Im(I)) \quad (3.5)$$

$$Q = (Re(I) * Im(V)) - (Im(I) * Re(V)) \quad (3.6)$$

Dónde:

P= potencia activa.
Q= potencia reactiva.
$Re(V)$=parte real de dela señal de voltaje.
$Re(I)$=parte real de dela señal de corriente.
$Im(V)$=parte imaginaria de la señal de voltaje.
$Im(I)$=parte imaginaria de la señal de corriente.

3.2.4 Tarea USB

Una vez terminadas la inicializaciones correspondientes a esta tarea, la tarea se bloquea esperando la activación por parte de la tarea "ADC". Realizado el desbloqueo la tarea ejecuta el grabado de los resultados en la memoria.

Para realizar el grabado de las variables calculadas en la memoria el procedimiento a seguir, es el que se detalla a continuación:

- *Abrir archivo donde se almacenarán los datos, especificando la ubicación y el nombre del archivo, si el proyecto es nuevo se crea este espacio en memoria, si el proyecto ya existe se empiezan a grabar los datos a partir del final establecido anteriormente.*

- *Una vez abierto el archivo, se transfieren los datos que se almacenaron en el buffer de resultado en el archivo abierto en el primer paso.*

- *Por último, es necesario cerrar el archivo creado.*

En la figura 3.10 se muestran los pasos a seguir en el almacenamiento de datos y la secuencia general de la tarea.

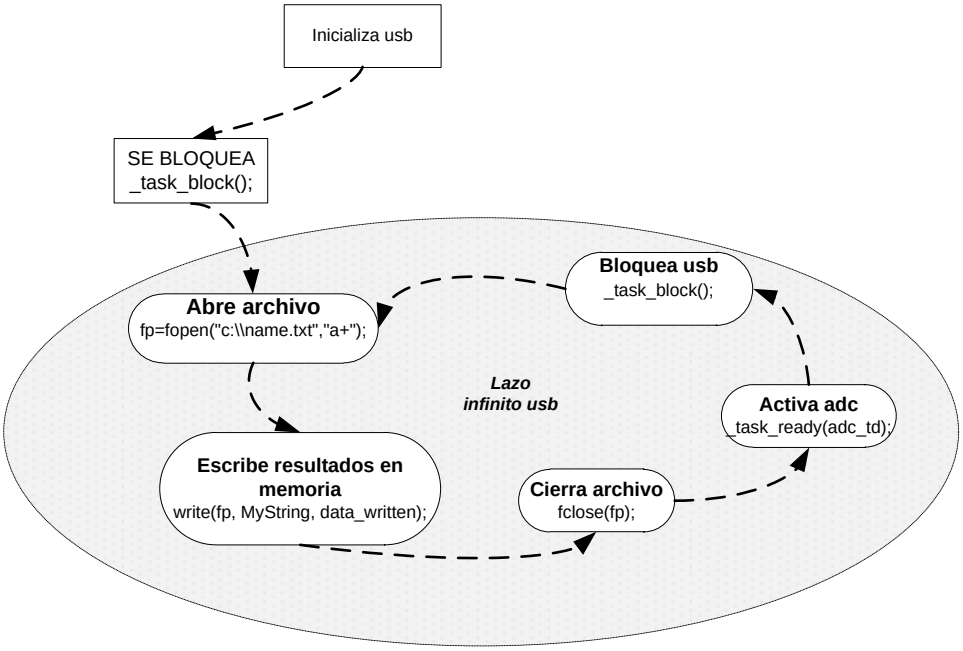


Figura 3.12. Diagrama de estados de la tarea USB.

3.3 Programación de la detección de oscilaciones de baja frecuencia

El programa realizado en MATLAB consta de dos fases, la primera es la que recibe los datos provenientes del microcontrolador y la segunda en donde se desarrolló el algoritmo de Prony para realizar la detección de oscilaciones de baja frecuencia.

3.3.1 Comunicación utilizando RS-232

La interfaz de comunicación fue programada en MATLAB y antecede al programa de Prony, en esta parte de comunicación se programa el enlace entre el microcontrolador y el programa de MATLAB, para este propósito es necesario saber en qué puerto de comunicación serial se encuentra el cable conectado en la computadora; en este caso se utiliza el COM 8 como puesto de comunicación serial, otro punto es saber el número de veces que cambia el estado de la señal transmitida en el tiempo, esta característica es medida en baudios, se utilizan 115,200 baudios por segundo, el tamaño del número a recibir (8 bits), y los bits de parada (1).

En la figura 3.11 se realiza una descripción general de la programación de la interfaz de comunicación y cómo se reciben los datos y son direccionados en variables que se utiliza como entrada al método de Prony.

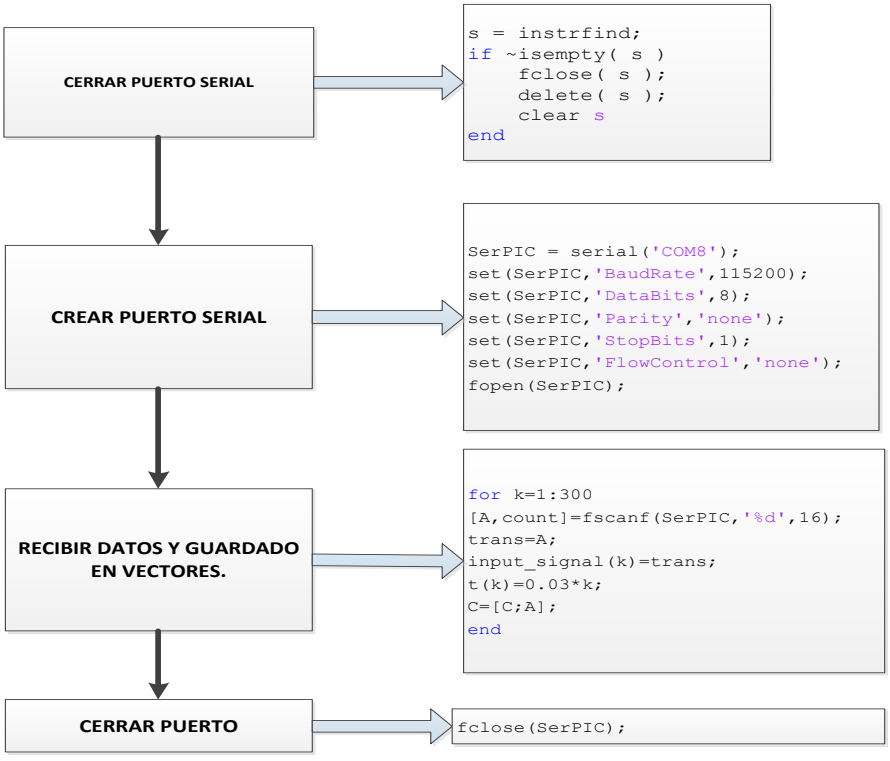


Figura 3.13. Secuencia de la interfaz de comunicación.

3.3.2 Programación del método de Prony

En la utilización del método de Prony es necesario tomar unas consideraciones previas para la señal de entrada con la cual se trabaja, esta señal debe estar acompañada de vector de tiempo de cada muestra y definir el periodo al cual estas señales están muestreadas. El programa de Prony empieza por formar la matriz de predicción lineal (ecuación 2.33), como ya se vio en el capítulo 2, esta matriz formada es no conformada, y se resuelve por medio de mínimos cuadrados, de esta manera se encuentran los coeficientes del polinomio de aproximación (ecuación 2.26). Una vez formado este polinomio, se obtienen las raíces con las cuales se calculan los eigenvalores (ecuación 2.34), en esta parte es importante recalcar que el número de eigenvalores encontrados es igual al valor del orden del polinomio de aproximación, en el programa implementado se filtran estos eigenvalores para sólo utilizar aquellos eigenvalores conjugados que se encuentren dentro del rango de las oscilaciones de baja frecuencia. Una vez obtenidas estas raíces se forma el modelo exponencial propuesto (ecuación 2.24) y se determina el valor de la amplitud compleja, de esta manera se obtiene nuestro modelo que aproxima a la señal de entrada (ecuación 2.21).

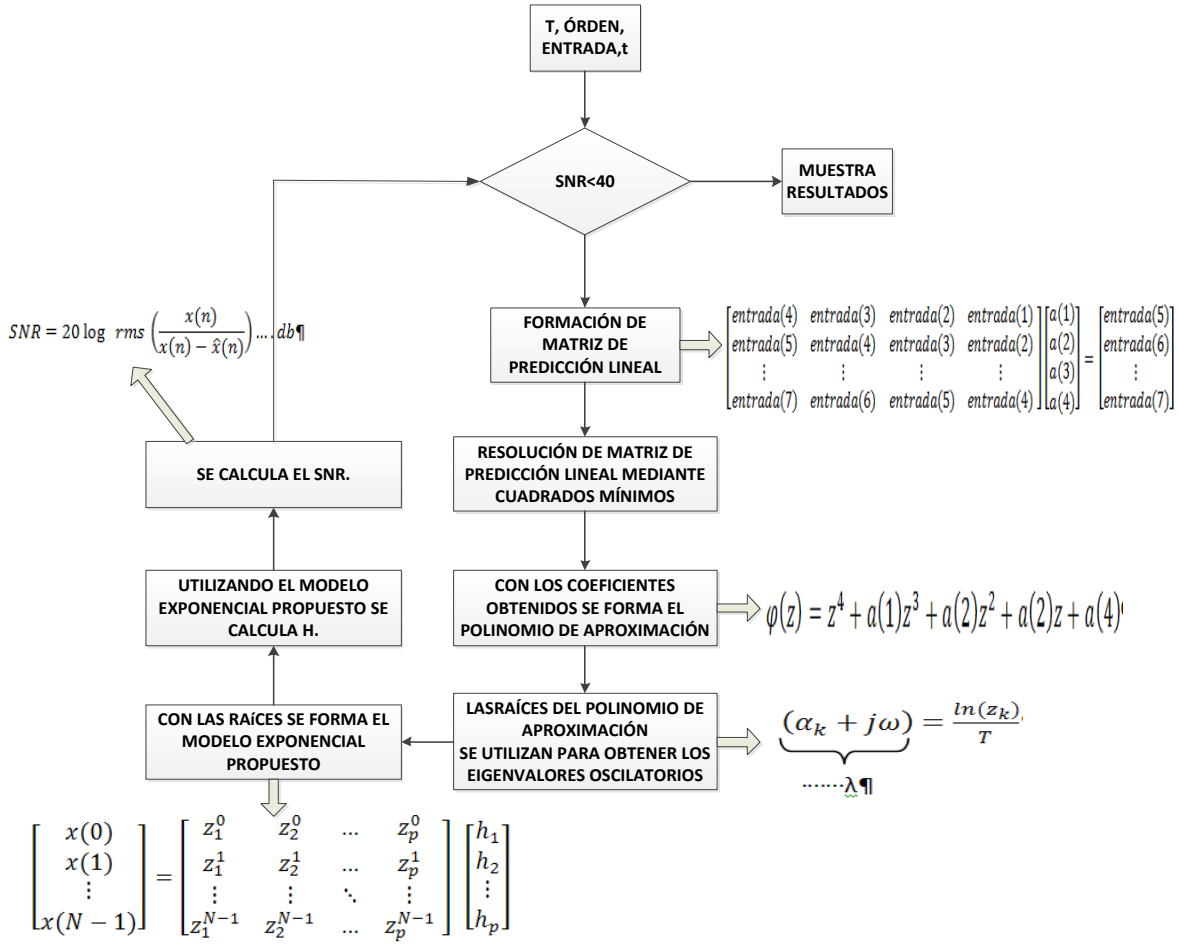


Figura 3.14. Diagrama de flujo del programa de Prony.

A continuación, se realiza un ejemplo con una señal de entrada conocida para ilustrar mejor el método de Prony implementado y llevar a cabo paso a paso el método programado con el objetivo de explicar de mejor manera el algoritmo programado en MATLAB.

Primero es necesario hacer unas observaciones iniciales:

- *Orden=4*
- *La señal de entrada utilizada se crea de acuerdo a lo revisado en las referencias [7][8][9] y muestra en según la forma programada en MATLAB:*
  - $input\_signal = 150 * exp(Df1 * t).* sin(f1 * t) + 160 * exp(Df2 * t).* sin(f2 * t);$
- *El tiempo de programación es t = [0:0.001:5]*

Para ejemplo utilizando $Df1 = -2,\ f1 = 15,\ Df2 = -3,\ f2 = 10,$ estos datos forman los siguientes eigenvalores:

$$\lambda_1 = -2 + 15i$$
$$\lambda_2 = -2 - 15i$$

$$\lambda_3 = -3 + 10i$$
$$\lambda_4 = -3 - 10i$$

- **Paso 1.**

El método de Prony logra desacoplar los valores de h_k y z_k utilizando solo 2p muestras de la señal de entrada, para las pruebas realizadas se utiliza un análisis con todos los valores obtenidos en nuestro intervalo de tiempo mencionado en las observaciones iniciales.

Siguiendo la referencia [1] y [4] utilizando el número máximo de muestras de Prony según la ecuación 2.33.

$$\begin{bmatrix} entrada(4) & entrada(3) & entrada(2) & entrada(1) \\ entrada(5) & entrada(4) & entrada(3) & entrada(2) \\ \vdots & \vdots & \vdots & \vdots \\ entrada(5000) & entrada(4999) & entrada(4998) & entrada(4997) \end{bmatrix} \begin{bmatrix} a(5) \\ a(2) \\ a(3) \\ a(4) \end{bmatrix} = \begin{bmatrix} entrada(5) \\ entrada(6) \\ \vdots \\ entrada(5001) \end{bmatrix}$$

- **Paso 2**.

Para este paso es necesario resolver la matriz formada en el paso anterior, con esta solución se obtiene el vector de coeficientes $\boldsymbol{a}$, para este paso se utiliza el método cuadrados mínimos como se muestra en la ecuación 3.7.

$$a = (X' * X)^{-1} * X' * b \tag{3.7}$$

Se forma el polinomio de aproximación $\varphi(z)$de la ecuación (2.26) de la siguiente manera obteniendo así la ecuación (3.8):

$$\varphi(z) = z^4 + a(1)z^3 + a(2)z^2 + a(2)z + a(4) \tag{3.8}$$

Según los resultados de nuestro ejemplo:

$$\varphi(z) = \boldsymbol{z^4} - 3.989688743396\ \boldsymbol{z^3} + 5.96942914683\ \boldsymbol{z^2} - 3.96979021529\ \boldsymbol{z} + 0.9900474548$$

Una vez obtenido este polinomio se procede a calcular las raíces, tomando en cuenta que las raíces complejas son aquellas con oscilaciones de las cuales es posible calcular los eigenvalores de la señal utilizando la ecuación (2.34):

$$\begin{aligned} z[1] &= 0.969192098810829 + 0.146479059078195i \\ z[2] &= 0.969192098810829 - 0.146479059078195i \\ z[3] &= 0.965597348042785 + 0.096882893454959i \\ z[4] &= 0.965597348042785 - 0.096882893454959i \end{aligned}$$

Y los eigenvalores calculados son:

$$\begin{aligned} \lambda_1 &= -2.000000000340843 + 14.999999988302905i \\ \lambda_2 &= -2.000000000340843 - 14.999999988302905i \\ \lambda_3 &= -2.999999999616888 + 10.000000017687121i \\ \lambda_4 &= -2.999999999616888 - 10.000000017687121i \end{aligned}$$

- **Paso 3.**

Para este paso es posible desarrollar el modelo exponencial propuesto, de donde se obtiene la amplitud compleja (h_k). La formulación del modelo se observa en la ecuación 3.9.

$$\begin{bmatrix} \text{input\_signal}(0) \\ \text{input\_signal}(1) \\ \vdots \\ \text{input\_signal}(4) \end{bmatrix} = \begin{bmatrix} z_1^0 & z_2^0 & \dots & z_p^0 \\ z_1^1 & z_2^1 & \dots & z_p^1 \\ \vdots & \vdots & \ddots & \vdots \\ z_1^{N-1} & z_2^{N-1} & \dots & z_p^{N-1} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_p \end{bmatrix} \tag{3.9}$$

Obteniendo los siguientes valores de h_k:

$$h_1 = 0.000000013383669 - 75.000000120744517i$$
$$h_2 = 0.000000013380031 + 75.000000120755431i$$
$$h_3 = -0.000000013380031 - 79.999999763749656i$$
$$h_4 = -0.000000013387307 + 79.999999763735104i$$

De esta manera se llega al fin del método, una variable que es importante a tomar en cuenta para la utilización del método de Prony es la amplitud de cada eigenvalor, se calcula como se muestra en la ecuación 3.10:

$$A_k = sqrt(imag(h).\hat{}2 + real(h).\hat{}2); \quad (3.10)$$

- **Paso 4.**

Se agrega este paso en donde se verifica si el modelo exponencial propuesto $\hat{x}(n)$ (de la ecuación 3.4), aproxima correctamente a la señal de entrada dada, utilizando a n como índice de cada valor desde $n = 0$ hasta $n = orden$:

Primero calculando:

$$\hat{x}(n) = Z * H$$

Y ahora si es posible calcular el SNR de la ecuación 3.37:

$$SNR = 20 \log \ rms \left(\frac{input\_signal(n)}{input\_signal(n) - \hat{x}(n)}\right)$$

CAPÍTULO 4

PRUEBAS Y ANÁLISIS DE RESULTADOS

En este capítulo se recopilan todas las pruebas realizadas al análisis de Prony programado en este trabajo, teniendo como objetivo principal la prueba final, donde interviene la implementación utilizando un simulador de sistemas de potencia en tiempo real y un microcontrolador.

Estas pruebas inician utilizando señales ideales sin ruido a diferentes valores de frecuencia de oscilación, probando cambios de orden y moviendo los eigenvalores; esto permitió tomar experiencia con el método y obtener más información acerca de la eficacia del algoritmo, después utilizando un sistema de potencia se realizan simulaciones para obtener valores de potencia activa, potencia reactiva y voltaje en terminales utilizando estas mediciones se calculan los eigenvalores oscilatorios validando resultados con paqueterías comerciales, por último se agrega la implementación utilizando el simulador que representa el final del proyecto.

4.1 Pruebas con señales ideales sin ruido

En esta parte se realiza la detección de oscilaciones a una señal programada en la cual se insertan los eigenvalores a encontrar, de esta manera tenemos una salida conocida, que nos permite sacar conclusiones básicas sobre el programa implementado.

4.1.1 Condiciones previas

Se utiliza la siguiente ecuación mostrada de la forma introducida en MATLAB con la cual se realizaron las pruebas con señales ideales.

$$150 * exp(Df1 * t).* sin(f1 * t) + 160 * exp(Df2 * t).* sin(f2 * t);$$

Donde $Df1$ y $Df2$ representan el factor de amortiguamiento de los eigenvalores introducidos, $f1$ y $f2$ representan los valores de frecuencia de los eigenvalores.

Antes de empezar el análisis de las pruebas, en cada una de las fases se muestra la tabla 4.1; para enfatizar por qué se decidió utilizar todas las muestras disponibles de la señal en lugar de sólo utilizar 2 veces el orden del polinomio de aproximación.

Tabla 4.1. Comparación método de Prony señal ideal utilizando diferentes valores de muestras para el análisis en el polinomio de aproximación.

Orden 4	Eigenvalores $(\alpha_k + j\omega)$	Frecuencia de oscilación (f_k)
Señal de entrada	$\mathbf{-2 + 15i}$ $\mathbf{-2 - 15i}$ $\mathbf{-3 + 10i}$ $\mathbf{-3 + 10i}$	2.387324146378430 -2.387324146378430 1.591549430918954 -1.591549430918954
Método de Prony (utilizando 2p muestras)	40.148903537913164 -2.716509243980154 +46.130759797896083i -2.716509243980154 -46.130759797896083i -44.718287876516889	- 7.341938450420044 7.341938450420044 -
Método de Prony (utilizando todas las muestras)	-2.021108889497766 +15.090128682169340i -2.021108889497766 -15.090128682169340i -2.978889993070167 + 9.861374689901375i -2.978889993070167 - 9.861374689901375i	2.401668571660039 -2.401668571660039 1.569486527579110 -1.569486527579110

4.1.2 Fase 1 (movimiento de eigenvalores en baja frecuencia)

4.1.2.1 Descripción de la prueba

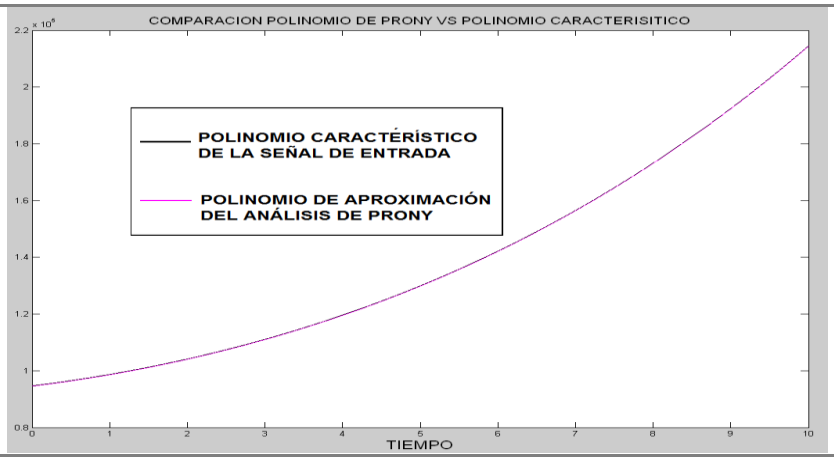
Se realizan movimientos en los eigenvalores de la entrada conocida, para observar el resultado obtenido del método Prony. Esto se realiza para evaluar la efectividad del algoritmo programado en las oscilaciones que se desean localizar (0.3hz -3.5hz).

Como ya se describió anteriormente, en este caso se introducen 4 eigenvalores en una señal de entrada programada en la cual se parte de eigenvalores con valores de frecuencia de oscilación ligeramente mayores a 3 Hz en donde primero se disminuye gradualmente un par de eigenvalores hasta llegar a frecuencias por debajo de los 0.5 Hz, desde ahí se empieza a disminuir el siguiente par de eigenvalores para encontrarse en el punto más bajo fijado para esta fase, en donde se utiliza el orden del polinomio igual a 4 que es el mismo orden de la señal de entrada. En la tabla 4.2 se muestran los resultados obtenidos en esta fase.

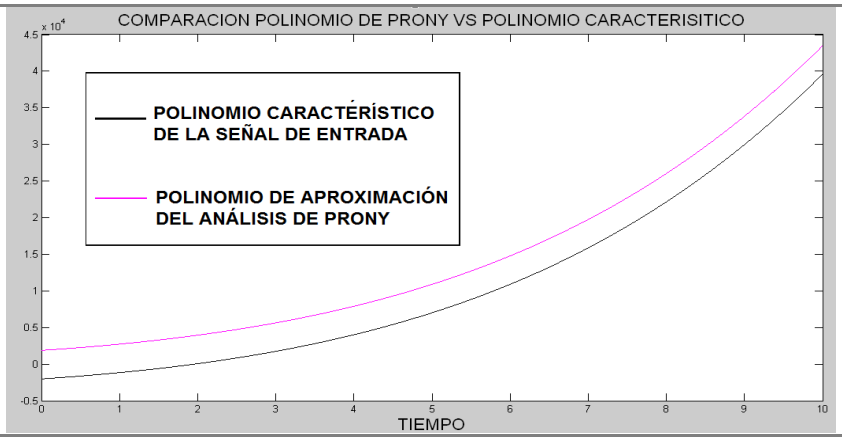
4.1.2.2 Resultados de la prueba

Tabla 4.2. Comportamiento de método de Prony ante diferentes entradas ideales.

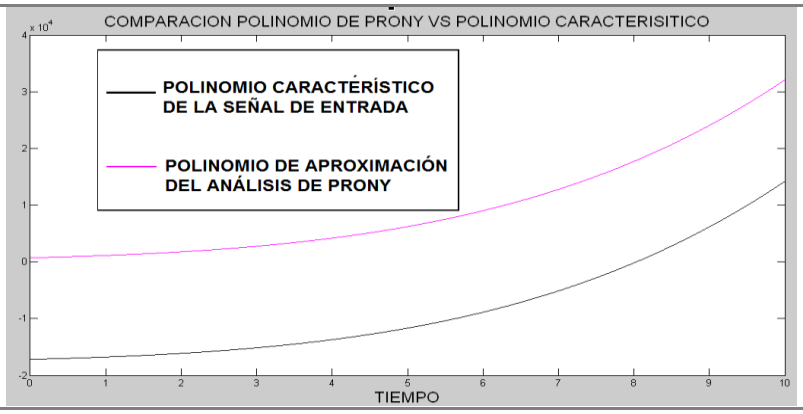
Señal de entrada de 4° orden		**Polinomio de Prony en 4° orden**		
Eigenvalores	Frecuencia de oscilación	Eigenvalores	Frecuencia de oscilación	SNR (signal to noise ratio)
$-2+20i$ $-2-20i$ $-3+15i$ $-3+15i$	3.1831 -3.1831 2.3873 -2.3873	-1.9955 +19.978i -1.9955 -19.978i -3.0045 +15.029i -3.0045 -15.029i	3.1796 -3.1796 2.392 -2.392	46.50811575537073
$-2+20i$ $-2-20i$ $-3+12i$ $-3+12i$	3.1831 -3.1831 1.9099 -1.9099	-1.995 +19.966i -1.995 -19.966i -3.005 +12.057i -3.005 -12.057i	3.1777 -3.1777 1.9189 -1.9189	44.37357125586742
$-2+20i$ $-2-20i$ $-3+10i$ $-3+10i$	3.1831 -3.1831 1.5915 -1.5915	-1.9944 +19.956i -1.9944 -19.956i -3.0056 +10.089i -3.0056 -10.089i	3.176 -3.176 1.6057 -1.6057	43.48369670221893
$-2+20i$ $-2-20i$ $-3+8i$ $-3+8i$	3.1831 -3.1831 1.2732 -1.2732	-1.9841 +19.868i -1.9841 -19.868i -3.0159 +8.3247i -3.0159 -8.3247i	3.1621 -3.1621 1.3249 -1.3249	40.91169822294338
$-2+20i$ $-2-20i$ $-3+6i$ $-3+6i$	3.1831 -3.1831 0.95493 -0.95493	-1.994 +19.951i 1.994 -19.951i -3.006 +6.1621i -3.006 -6.1621i	3.1753 -3.1753 0.98073 -0.98073	42.40822725145674
$-2+20i$ $-2-20i$ $-3+4i$ $-3+4i$	3.1831 -3.1831 0.63662 -0.63662	-1.995 +19.949i -1.995 -19.949i -3.005 +4.2478i -3.005 -4.2478i	3.175 -3.175 0.67605 -0.67605	42.65882622421673
$-2+20i$ $-2-20i$ $-3+3i$ $-3+3i$	3.1831 -3.1831 0.47746 -0.47746	-1.9889 +19.874i -1.9889 -19.874i -3.0111 +3.7496i -3.0111 -3.7496i	3.163 -3.163 0.59676 -0.59676	41.11029905620810
$-2+15i$ $-2-15i$ $-3+3i$ $-3+3i$	2.3873 -2.3873 0.47746 -0.47746	-1.9905 +14.968i -1.9905 -14.968i -3.0095 +3.1589i -3.0095 -3.1589i	2.3822 -2.3822 0.50275 -0.50275	42.59953347385556
$-2+12i$ $-2-12i$ $-3+3i$ $-3+3i$	1.9099 1.9099 0.47746 -0.47746	0.25595 +9.8087i 0.25595 -9.8087i -5.2559 +8.4547i -5.2559 -8.4547i	1.5611 -1.5611 1.3456 -1.3456	38.04900358026424
$-2+10i$ $-2-10i$ $-3+3i$ $-3+3i$	1.5915 -1.5915 0.47746 -0.47746	-2.1958 +11.463i -2.1958 -11.463i 1.9629 -7.5713	1.8245 -1.8245 0 0	39.65122876509353
$-2+8i$ $-2-8i$ $-3+3i$ $-3+3i$	1.2732 -1.2732 0.47746 -0.47746	-2.2999 +9.6549i -2.2999 -9.6549i 1.842 -7.2422	1.5366 -1.5366 0 0	38.81448959135097
$-2+6i$ $-2-6i$ $-3+3i$ $-3+3i$	0.63662 -0.63662 0.47746 -0.47746	8.0378 -2.446 +12.518i -2.446 -12.518i -13.146	0 1.9923 -1.9923 0	36.26558394820322
$2+4i$ $-2-4i$ $-3+3i$ $-3+3i$	0.95493 -0.95493 0.47746 -0.47746	8.8252 +11.932i 8.8252 -11.932i -13.825 +11.772i -13.825 -11.772i	1.899 -1.899 1.8736 -1.8736	32.94478606569294



$-2+20i$
$-2-20i$
$-3+15i$
$-3+15i$



$-2+12i$
$-2-12i$
$-3+3i$
$-3+3i$



$-2+6i$
$-2-6i$
$-3+3i$
$-3+3i$

Figura. 4.1. Comparación de los polinomios característicos de señales de entrada con los polinomios obtenidos del método de Prony de la fase 1.

4.1.2.3 Análisis de la prueba

En la tabla 4.2 y la figura 4.1 se observa el comportamiento del método ante señales ideales; cuando se trabaja con este tipo de entrada podemos describir el comportamiento del método de forma más sencilla, se marcaron con diferentes tonos de sombreado en la tabla 4.2 valores en los cuales el método con un orden fijo de 4 no detecta de manera precisa los eigenvalores de la entrada, esto se corrobora con la figura 4.1 donde se observa que a medida que la señal del polinomio característico tiende a ser más lenta el polinomio de aproximación del algoritmo de Prony se separa del polinomio característico de la señal de entrada, lo que nos entrega como resultado eigenvalores oscilatorios erróneos

A medida que las oscilaciones introducidas en la señal de entrada tienden a ser más lentas el método de Prony necesita mayor número de muestras en la matriz de predicción lineal, lo cual se puede conseguir aumentando el orden del polinomio de aproximación, en la figura 4.2 se observa la diferencia entre dos señales de entrada, la primer señal introducida en la prueba de la fase 1 y el caso critico final.

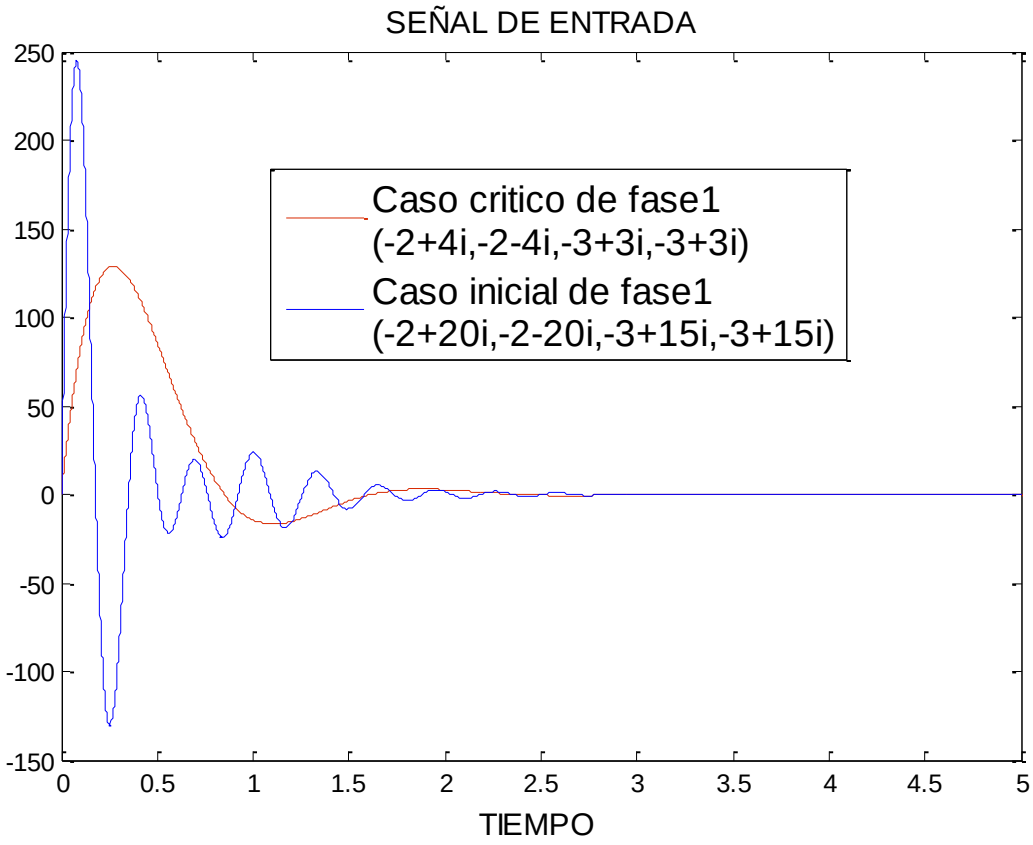


Figura. 4.2. Comparación de las señales de entrada caso inicial y caso crítico.

4.1.3 Fase 2 (Comparación moviendo el orden del polinomio de aproximación)

4.1.3.1 Descripción de la prueba

Se realiza una comparación moviendo el orden del polinomio de aproximación de Prony, conservando la misma señal de entrada y así observar el comportamiento en órdenes diferentes del método.

Para esta prueba se utilizan casos denominados críticos y sombreados en la tabla 4.2 donde el valor de SNR fue menor a 40 db y se prueban estos valores de entrada moviendo el orden, para probar que a medida que las oscilaciones son más lentas se tiene que aumentar el número de datos a tomar en cuenta lo cual se consigue aumentando el orden del polinomio.

Para esta fase se analizaron 5 valores críticos mostrando en las tablas los eigenvalores a detectar de manera resaltada para diferenciarlos de los eigenvalores accesorios que se agregan al aumentar el orden del polinomio.

4.1.3.2 Resultados de la prueba

Tabla 4.3. Comparación variando el orden en el algoritmo de Prony ante una señal de entrada con eigenvalores $-2+12i, -2-12i, \ -3+3i, -3+3i.$

	Señal de entrada	4° orden *(SNR= 38.0490035)*	12° orden *(SNR= 48.6070889)*	24° orden *(SNR= 52.355541)*		30°orden *(SNR= 51.52288436)*	
Eigenvalores	$\mathbf{-2+12i}$ $\mathbf{-2-12i}$ $\mathbf{-3+3i}$ $\mathbf{-3+3i}$	0.25595 +9.8087i 0.25595 -9.8087i -5.2559 +8.4547i -5.2559 -8.4547i	-151.95 +2875.9i -151.95 -2875.9i -146.74 +2342.4i -146.74 -2342.4i -135.44 +1800.3i -135.44 -1800.3i -115.14 +1231.9i -115.14 -1231.9i -2 +12i -2 -12i -3 +3.0013i -3 -3.0013i	-2+12i -2 -12i -3 +3i -3 -3i -59.861 +619.87i -59.861 -619.87i -72.281 +905.23i -72.281 -905.23i -80.944 +1176.8i -80.944 -1176.8i -87.446 +1443.1i -87.446 -1443.1i	-92.448 +1706.8i -92.448 -1706.8i -96.298 +1969i -96.298 -1969i -99.203 +2230.3i -99.203 -2230.3i -101.29 +2491i -101.29 -2491i -103.31 +3011.5i -103.31 -3011.5i -102.64 +2751.4i -102.64 -2751.4i	-2 +12i -2 -12i -3 +3i -3 -3i -48.186+497.13i -48.186 497.13i -58.36 + 725.94i -58.36 -725.94i -65.587+943.68i -65.587-943.68i -71.153+1157.2i -71.153-1157.2i -75.591+1368.6i -75.591-1368.6i -79.185+1578.7i -79.185-1578.7i	-82.109+1788.1i -82.109-1788.1i -84.476+1996.9i -84.476-1996.9i -86.361+2205.4i -86.361-2205.4i -89.922+3037.6i -89.922-3037.6i -89.578+2829.7i -89.578-2829.7i -88.881+2621.8i -88.881-2621.8i -87.817+2413.7i -87.817-2413.7i

Tabla 4.4. Comparación variando el orden en el algoritmo de Prony ante una señal de entrada con eigenvalores $-2+10i, -2-10i, \ -3+3i, -3+3i.$

	Señal de entrada	4° orden *(SNR= 39.6512287)*	12° orden *(SNR= 50.8016028)*	24° orden *(SNR= 56.514835912)*		30°orden *(SNR= 60.174324563)*	
Eigenvalores	$-2+10i$ $-2-10i$ $-3+3i$ $-3+3i$	-2.1958 +11.463i -2.1958 -11.463i 1.9629 -7.5713	-151.95 +2875.9i -151.95 -2875.9i -146.74 +2342.4i -146.74 -2342.4i -135.44 +1800.3i -135.44 -1800.3i -115.14 +1231.9i -115.14 -1231.9i -2 +12i -2 -12i -3 +3.0013i -3 -3.0013i	-2+12i -2 -12i -3 +3i -3 -3i -59.861 +619.87i -59.861 -619.87i -72.281 +905.23i -72.281 -905.23i -80.944 +1176.8i -80.944 -1176.8i -87.446 +1443.1i -87.446 -1443.1i	-92.448 +1706.8i -92.448 -1706.8i -96.298 +1969i -96.298 -1969i -99.203 +2230.3i -99.203 -2230.3i -101.29 +2491i -101.29 -2491i -103.31 +3011.5i -103.31 -3011.5i -102.64 +2751.4i -102.64 -2751.4i	-2 +12i -2 -12i -3 +3i -3 -3i -48.186+497.13i -48.186 497.13i -58.36 + 725.94i -58.36 -725.94i -65.587+943.68i -65.587-943.68i -71.153+1157.2i -71.153-1157.2i -75.591+1368.6i -75.591-1368.6i -79.185+1578.7i -79.185-1578.7i	-82.109+1788.1i -82.109-1788.1i -84.476+1996.9i -84.476-1996.9i -86.361+2205.4i -86.361-2205.4i -89.922+3037.6i -89.922-3037.6i -89.578+2829.7i -89.578-2829.7i -88.881+2621.8i -88.881-2621.8i -87.817+2413.7i -87.817-2413.7i

Tabla 4.5. Comparación variando el orden en el algoritmo de Prony ante una señal de entrada con eigenvalores $-2+8i, -2-8i, \ -3+3i, -3+3i.$

	Señal de entrada	4° orden *(SNR= 38.8144895)*	12° orden *(SNR= 47.357274)*	24° orden *(SNR= 52.5780472)*		30°orden *(SNR= 50.744698689)*	
Eigenvalores	$-2+8i$ $-2-8i$ $-3+3i$ $-3+3i$	-2.2999 +9.6549i -2.2999-9.6549i 1.842 -7.2422	-151.96 +2875.9i -151.96 -2875.9i -146.75 +2342.4i -146.75 -2342.4i -135.44 +1800.3i -135.44 -1800.3i -115.14 +1231.9i -115.14 -1231.9i -2.0017 +8.0063i -2.0017 -8.0063i -2.9983 +2.9825i -2.9983 -2.9825i	-2 +8i -2 8i -3 +2.9999i -3 -2.9999i -59.872 +619.84i -59.872 -619.84i -72.291 +905.21i -72.291 -905.21i -80.953 +1176.8i -80.953 -1176.8i -87.455 +1443.1i -87.455 - 1443.1i -92.457 +1706.8i -92.457 -1706.8i -96.307 + 1969i -96.307 - 1969i	-99.211 +2230.3i -99.211 -2230.3i -101.3 + 2491i -101.3 -2491i -103.32 +3011.5i -103.32 - 3011.5i -102.65 +2751.4i -102.65 -2751.4i	-2 +8i -2 -8i -3 +3i -3 -3i -48.199+497.09i -48.199-497.09i -58.372+725.91i -58.372-725.91i -65.599+943.66i -65.599-943.66i -71.164+1157.2i -71.164-1157.2i -75.601+1368.6i -75.601-1368.6i -79.196+1578.7i -79.196-1578.7i	-82.12+1788.1i -82.12-1788.1i -84.486+1996.9i -84.486-1996.9i -86.371+2205.4i -86.371-2205.4i -89.932+3037.6i -89.932-3037.6i -89.588+2829.7i -89.588-2829.7i -88.891+2621.8i -88.891-2621.8i -87.827+2413.7i -87.827-2413.7i

Tabla 4.6. Comparación variando el orden en el algoritmo de Prony ante una señal de entrada con eigenvalores $-2+6i, -2-6i, \ -3+3i, -3+3i$.

	Señal de entrada	4° orden *(SNR= 3.6265583)*	12° orden *(SNR= 47.014846)*	24° orden *(SNR= 51.40083588)*		30°orden *(SNR= 47.939615)*	
Eigenvalores	$-2+6i$ $-2-6i$ $-3+3i$ $-3+3i$	8.0378 -2.446 +12.518i -2.446 - 12.518i -13.146	-151.96+2875.9i -151.96 -2875.9i -146.75 +2342.4i -146.75 -2342.4i -135.45 +1800.3i -135.45 -1800.3i -115.14 +1231.9i -115.14 - 1231.9i -1.9958 +5.9916i -1.9958 -5.9916i -3.0042 +3.0181i -3.0042 -3.0181i	-2 +6.0001i -2 -6.0001i -3 +2.9998i -3 - 2.9998i -59.875+619.83i -59.875 -619.83i -72.294 +905.21i -72.294 -905.21i -80.956 +1176.8i -80.956 -1176.8i -87.458 +1443.1i -87.458 -1443.1i -92.46 +1706.8i -92.46 -1706.8i	-96.31 +1969i -96.31 -1969i -99.214 +2230.3i -99.214 -2230.3i -101.3 +2491i -101.3 -2491i -103.32 +3011.5i -103.32 -3011.5i -102.65 + 2751.4i -102.65 -2751.4i	-2+6.0001i -2 -6.0001i -3 +2.9998i -3 -2.9998i -48.204+497.07i -48.204-497.07i -58.376+725.9i -58.376-725.9i -65.602+943.66i -65.602-943.66i -71.168+1157.2i -71.168-1157.2i -75.605+1368.5i -75.605-1368.5i -79.199+1578.7i -79.199-1578.7i	-82.123+1788.1i -82.123-1788.1i -84.49+1996.9i -84.49-1996.9i -86.375+2205.4i -86.375-2205.4i -89.936+3037.6i -89.936-3037.6i -89.591+2829.7i -89.591-2829.7i -88.895+2621.7i -88.895-2621.7i -87.831+2413.7i -87.831-2413.7i

Tabla 4.7. Comparación variando el orden en el algoritmo de Prony ante una señal de entrada con eigenvalores $-2+4i, -2-4i, \ -3+3i, -3+3i$.

	Señal de entrada	4° orden *(SNR= 32.944786)*	12° orden *(SNR= 45.541399)*	24° orden *(SNR= 48.0077339)*		30°orden *(SNR= 46.49569679)*	
Eigenvalores	$-2+4i$ $-2-4i$ $-3+3i$ $-3+3i$	8.8252 +11.932i 8.8252 -11.932i -13.825 +11.772i -13.825 -11.772i	-151.96+2875.9i -151.96 -2875.9i -146.75 +2342.4i -146.75 -2342.4i -135.45 +1800.3i -135.45 - 1800.3i -115.15 + 1231.9i -115.15 -1231.9i -1.9942 +4.0107i -1.9942 -4.0107i -3.0058 + 2.9878i -3.0058 - 2.9878i	-2.0073+4.0057i -2.0073 -4.0057i -2.9927 +2.9899i -2.9927 -2.9899i -59.878 +619.82i -59.878 -619.82i -72.296 +905.2i -72.296 - 905.2i -80.958 +1176.8i -80.958 -1176.8i -87.46 +1443.1i -87.46 -1443.1i -92.462+1706.8i -92.462-1706.8i -96.312 +1969i -96.312 -1969i	-99.216+2230.3i -99.216 -2230.3i -101.3 +2491i -101.3 - 2491i -103.32 +3011.5i -103.32 -3011.5i -102.66 + 2751.4i -102.66 - 2751.4i	-1.9995+3.9997i -1.9995-3.9997i -3.0005+3.0006i -3.0005-3.0006i -48.208+497.07i -48.208-497.07i -58.379+725.9i -58.379 - 725.9i -65.605+943.65i -65.605-943.65i -71.171+1157.2i -71.171-1157.2i -75.608+1368.5i -75.608-1368.5i -79.202+1578.7i -79.202-1578.7i	-82.126+1788.1i -82.126-1788.1i -84.492+1996.9i -84.492-1996.9i -86.377+2205.4i -86.377-2205.4i -89.939+3037.6i -89.939-3037.6i -89.594+2829.7i -89.594-2829.7i -88.897+2621.7i -88.897-2621.7i -87.833+2413.7i -87.833-2413.7i

4.1.3.3 Análisis de la prueba

Al aumentar el orden del polinomio, se consiguió mejorar la detección de eigenvalores oscilatorios de la señal de entrada; aunque al ejecutar este procedimiento se agrega al resultado eigenvalores espurios, los cuales no pueden ser desechados del resultado; para lo cual es importante encontrar algún parámetro qué indique que eigenvalores conjugados son los modos característicos de la señal de entrada y cuáles son aquellos que forman parte del ruido, esta propiedad se retomará posteriormente al analizar la amplitud como parámetro para descartar modos accesorios en el análisis del Prony.

En la figura 4.3 se muestra la señal de entrada comparada con el algoritmo de Prony propuesto a diferentes valores de orden, esta gráfica se elabora utilizando el modelo exponencial propuesto y se puede observar que conforme el orden de polinomio del algoritmo de Prony es aumentado la aproximación a la señal de entrada mejora. Es por esto que en la tabla 4.7 se observa que a partir del orden 12, el método de Prony ajusta la señal de entrada sin error.

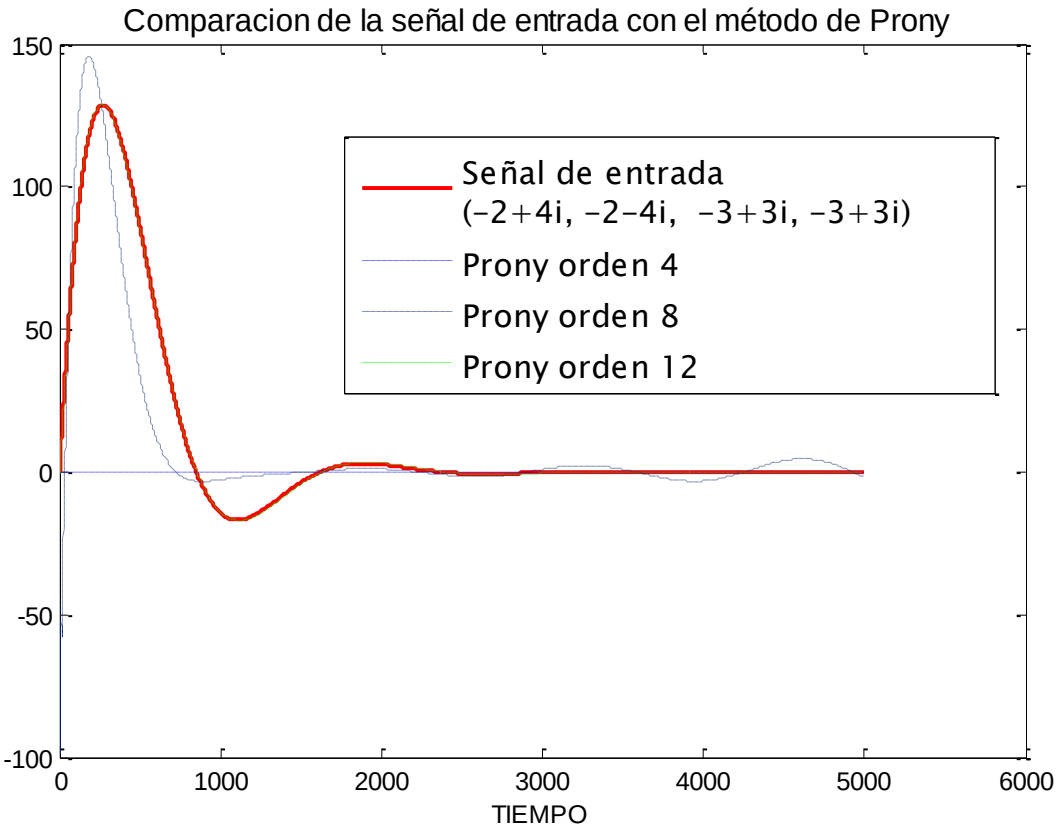


Figura. 4.3. Comparación de la señal de entrada caso crítico con el método de Prony a diferentes órdenes de aproximación.

4.1.4 Fase 3 (Análisis de tiempo de muestreo)

4.1.4.1 Descripción de la prueba

Para esta fase se utilizan eigenvalores de entrada menores que el caso crítico, de acuerdo a lo visto en las fases anteriores se necesita en este tipo de eigenvalores mayor orden para detectarlos.

Los eigenvalores utilizados en esta fase, se encontraron como los más conflictivos para el análisis con señales ideales.

El objetivo de esta fase es determinar la importancia del tiempo de muestreo en el análisis de las señales más conflictivas de la entrada.

4.1.4.2 Resultados de la prueba

Tabla 4.8. Comparación de diferentes tiempos de muestreo para caso especial.

Señal de entrada eigenvalores	**12° orden**		
	0.001	0.01	0.1
$-0.028 + 2.8903i$ $-0.028 - 2.8903i$ $-0.0634 + 3.1416i$ $-0.0634 - 3.1416i$	-0.094906±8.0317i	-0.063402 ± 3.1416i -0.027998 ± 2.8903i	-1.2514 ± 18.181i -1.0371 ± 12.635i -0.0634 ± 3.1416i -0.028 ± 2.8903i
	24°orden		
	0.001	0.01	0.1
	-0.041841 ± 3.4183i -0.049561 ± 2.5568i	-0.0634 ± 3.1416i -0.028 ± 2.8903i	-0.028 ± 2.8903i -0.0634 ± 3.1416i -0.94512 ± 7.29i -1.2236 ± 10.051i -1.4223 ± 12.74i -1.5781 ± 15.406i -1.7055 ± 18.065i -1.8108 ± 20.723i

4.1.4.3 Análisis de la prueba

En esta fase se analizaron 3 tiempos de muestreo, moviendo el orden de 12 a 24, se puede observar que para este caso en específico cuando se analiza la señal de entrada especial para detectar las oscilaciones de baja frecuencia, se observa que cuando se disminuye el número de muestras por ciclo se aumenta la rapidez con la que la señal muestra su comportamiento completo.

Al mostrar su comportamiento oscilatorio más rápidamente, es posible utilizar un menor orden del polinomio de aproximación para encontrar los eigenvalores introducidos, una observación importante de esta prueba es que resalta la importancia de contar con un muestreo adecuado de la señal.

El tiempo de muestreo de 0.1 segundos, encuentra los eigenvalores de la entrada de manera exacta con menor orden que los otros tiempos de muestreo; pero añade modos espurios en el rango de las oscilaciones de baja frecuencia, para lo que es necesario tomar en cuenta la amplitud; para descartar estos valores de los representativos de la señal de entrada.

Lo mostrado en la tabla 4.8 se puede observar de manera gráfica en las figuras 4.4 y 4.5 en donde se observa que la mejor aproximación de la señal de entrada se obtiene a un tiempo de muestreo de 0.01 segundos.

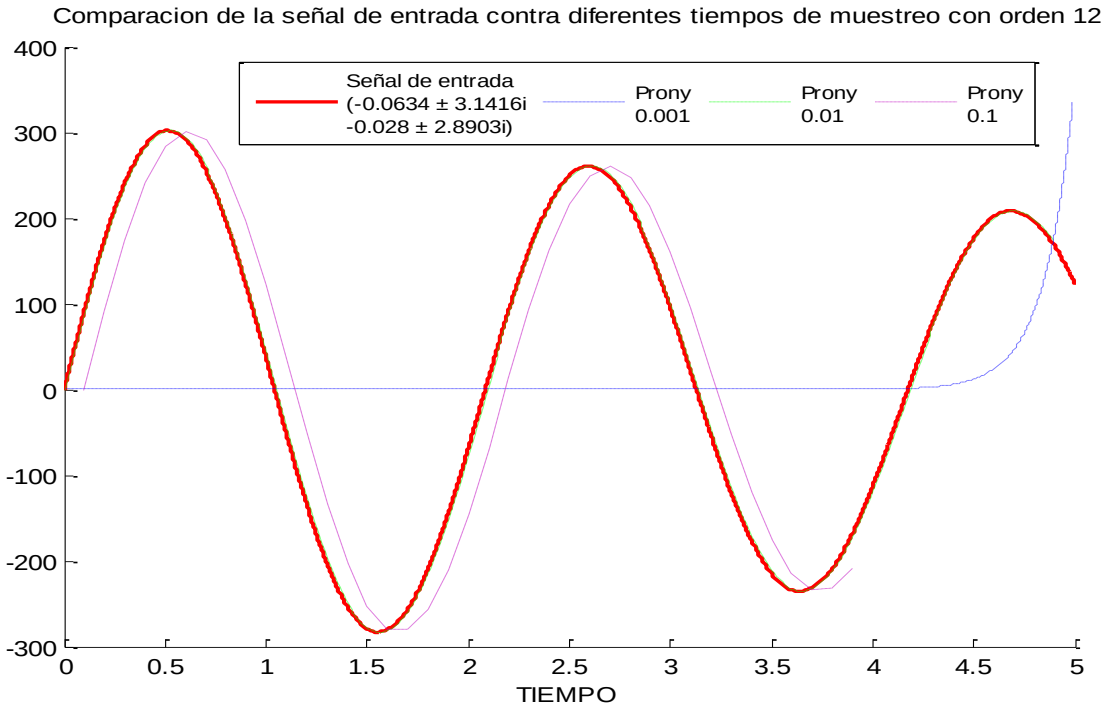


Figura 4.4. Comparación de la señal de entrada para el caso especial contra el algoritmo de Prony a diferentes tiempos de muestreo con orden 12.

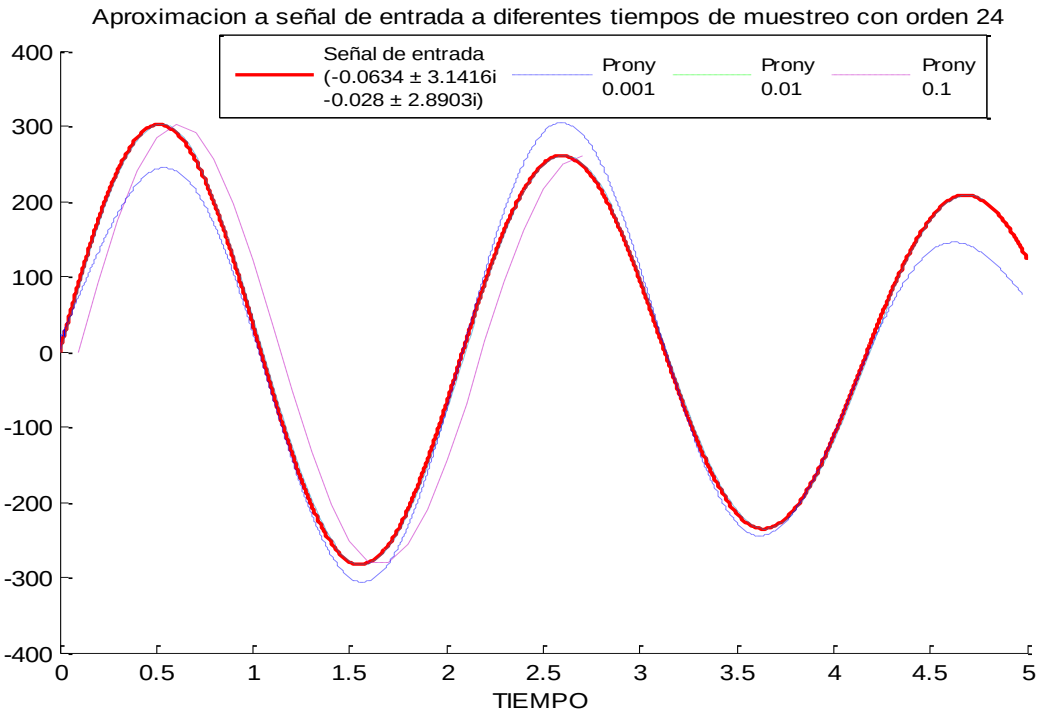


Figura 4.5. Comparación de la señal de entrada para el caso especial contra el algoritmo de Prony a diferentes tiempos de muestreo con orden 24.

4.2 Prueba con sistema de potencia de dos áreas

4.2.1 Descripción de la prueba

Para realizar esta prueba se utiliza el sistema de potencia de dos áreas que presenta oscilaciones de baja frecuencia, las cuales han sido muy estudiadas anteriormente.
Este sistema se cargó utilizando los datos mostrados en el apéndice A, el sistema es simulado en un programa especializado de sistemas de potencia con el cual se analizan diferentes variables como son: potencia activa, potencia reactiva y el voltaje en terminales en los nodos de generación de la red, estas variables son utilizadas como señales entrada para el algoritmo de Prony ya descrito, los datos de salida obtenidos con el método de Prony son comparados con la literatura especializada. Los resultados de los paquetes comerciales fueron obtenidos de [24] y [6], también se realiza una comparación con [11].

Los paquetes utilizados adaptados de [6]:

PSS/E [25]
DIgSILENT[26]
EUROSTAG[27]

También se realizaron simulaciones dinámicas y cálculo de eigenvalores en The Power System Analysis Toolbox (PSAT) [28] para MATLAB Y en Small Signal Analysis Tool (SSAT) [29] de Power Tech.

En la figura 4.6 se muestran los puntos de análisis en el sistema de potencia de dos áreas de los cuales se obtuvieron las mediciones de las variables mencionadas utilizando el PSAT para MATLAB. Se realiza una simulación dinámica ante una falla trifásica en el nodo 8 del sistema de potencia mostrado en la figura 4.6. Se miden las variables mencionadas de 0 a 20 segundos tomando ventanas de esta señal como entrada para el algoritmo de Prony. Se comparan los resultados del análisis de la estabilidad de pequeña señal en PSAT y una adaptación de las pruebas realizadas con el mismo sistema en [6].

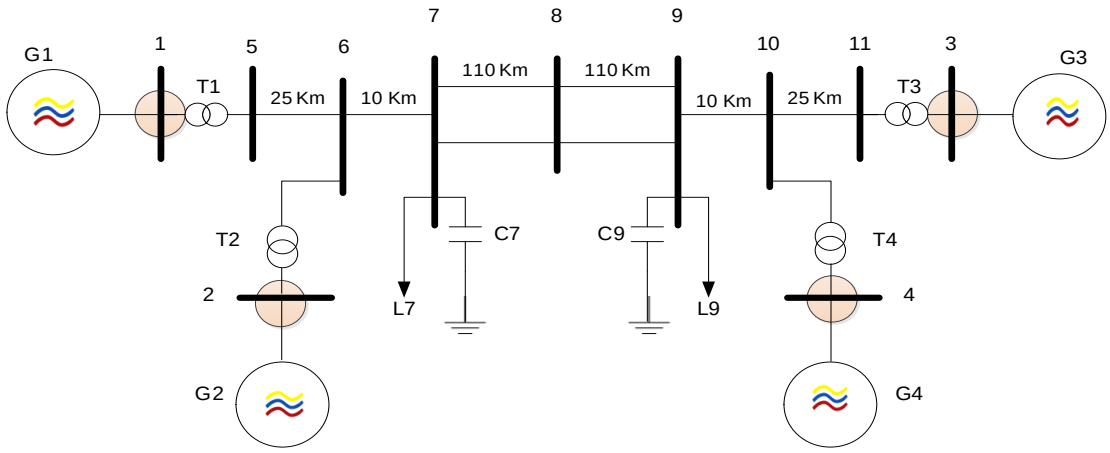


Figura 4.6. Puntos a analizar del sistema de potencia de dos áreas [11].

En las figuras 4.7, 4.8 y 4.9 se muestra el manejo de la simulación realizada en PSAT obteniendo el comportamiento de las variables durante un periodo completo de 20 segundos de los cuales se analizaron 10 segundos con un muestreo de 0.01 segundos.

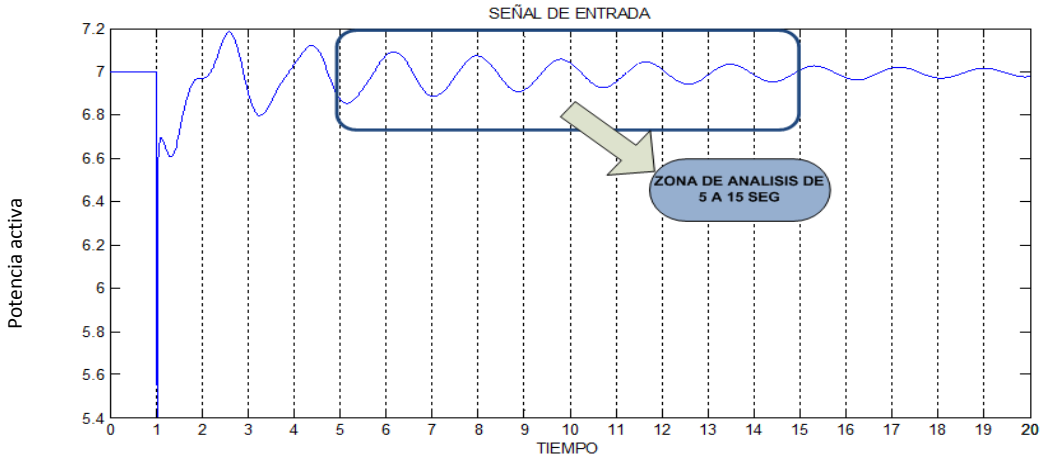


Figura 4.7. Señal de entrada de la potencia activa especificando el área de análisis (nodo 1).

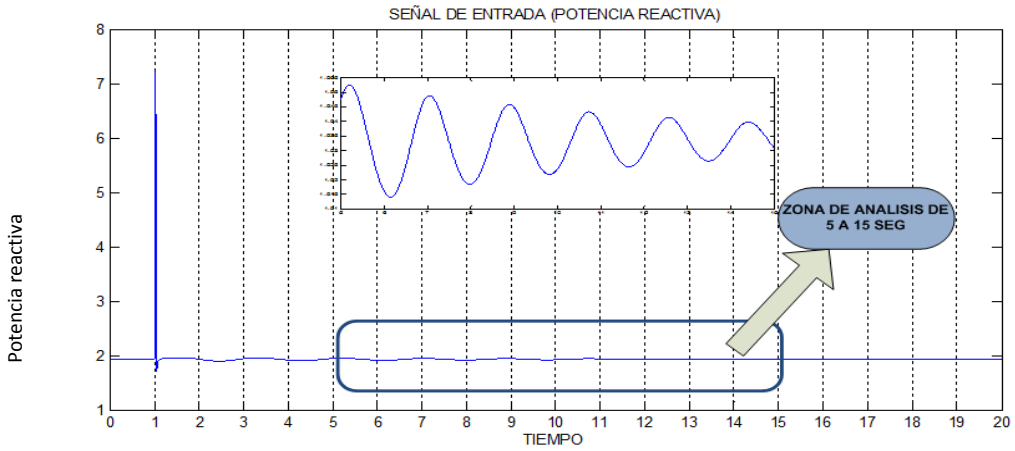


Figura 4.8. Señal de entrada para la potencia reactiva especificando el área de análisis (nodo 1).

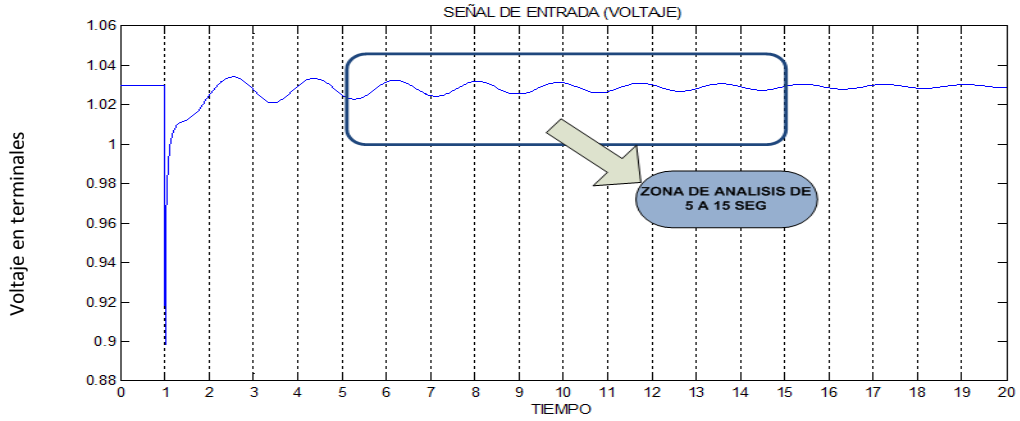


Figura 4.9. Señal de entrada para el voltaje especificando el área de análisis (nodo 1).

En las tablas 4.9 y 4.10 se puede observar los resultados obtenidos al analizar la señal con el programa PSAT. En cada tabla se especifica el orden del polinomio de aproximación el cual se obtuvo partiendo de un orden de 50 hasta llegar al orden donde el valor de la relación señal ruido (SNR) supera los 40 db. También se muestran los valores de la frecuencia (f) y amplitud (A) de cada eigenvalor oscilatorio. En las tablas se muestran solo los eigenvalores oscilatorios en el rango de las oscilaciones de baja frecuencia (0.5 a 3Hz).

Se realzan los eigenvalores similares a los encontrados por el PSAT para hacer más fácil la visualización de los resultados, se especifican los eigenvalores encontrados por nodo de análisis según lo observado en la figura 4.6.

4.2.2 Resultados de la prueba

Tabla 4.9. Análisis de resultados obtenidos con el algoritmo de Prony y el programa PSAT prueba con potencia activa y potencia reactiva.

Eigenvalores calculados PSAT	Eigenvalores calculados con el algoritmo de Prony			
	Nodo 1	Nodo 2	Nodo 3	Nodo 4
-0.5554± j6.594 (f = 1.0495 Hz) -0.5664± j6.785 (f = 1.0799Hz) -0.1307± j3.413 (f 0.54329Hz)	Potencia activa (orden 54) **-0.67681 ± 6.7922i** (f = 1.081Hz) (A=0.0074131) **-0.12923 ± 3.4164i** (f = 0.54374Hz) (A=0.063893) SNR=48.148 db	Potencia activa (orden 66) **-0.56955 ± 6.7289i** (f = 1.0709 Hz) (A=0.0055873) **-0.12382 ± 3.4091i** (f = 0.54257 Hz) (A=0.024002) **-10.668 ± 14.6i** (f = 2.3236Hz) (A=0.00017234) SNR=43.24 db	Potencia activa (orden 52) **-0.43096 ± 6.4999i** (f = 1.0345 Hz) (A=0.0044396) **-0.12503 ± 3.418i** (f= 0.54399Hz) (A=0.032638) SNR=40.401db	Potencia activa (orden 56) **-0.68781 ± 6.7275i** (f = 1.0707 Hz) (A= 0.005787) **-0.12979 ± 3.4176i** (f= 0.54393 Hz) (A= 0.06913) SNR=43.72db
-0.5554± j6.594 (f = 1.0495 Hz) -0.5664± j6.785 (f = 1.0799Hz) -0.1307± j3.413 (f 0.54329Hz)	Potencia Reactiva (orden 214) **-0.22574±6.8065i** (f = 1.0833 Hz) (A=0.00077909) **-0.13001±3.4189i** (f = 0.54414Hz) (A=0.010582) **-0.65633±6.9514i** (f = 1.1063 Hz) (A=0.00054337) **-0.57849±10.138i** (f = 1.6136 Hz) (A=1.9814e-005) **-0.75829±15.667i** (f = 2.4935Hz) (A=1.3556e-006) **-0.75491±20.482i** (f = 3.2598 Hz) (A=1.3126e-006) **-3.9828±19.767i** (f = 3.146 Hz) (A=1.5808e-005) SNR=40.485	Potencia Reactiva (orden 106) **-0.56491±6.6756i** (f = 1.0625Hz) (A=0.0012907) **-0.12945±3.4184i** (f = 0.54406Hz) (A=0.061253) **-1.4672±9.9246i** (f = 1.5795 Hz) (A=9.7078e-005) SNR=40.344	Potencia Reactiva (orden 146) **-0.12942±3.4186i** (f = 0.54409Hz) (A=0.0026569) **-0.4384 ±6.815i** (f= 1.0846 Hz) (A=4.8901e-005) **-1.4793±14.788i** (f = 2.3536 Hz) (A= 8.0266e-007) **-1.3584±18.297i** (f= 2.912 Hz) (A=1.5649e-006) SNR=41.97	Potencia Reactiva (orden 62) **-0.49566±6.6406i** (f = 1.0569 Hz) (A= 0.0022558) **-0.15227±3.4015i** (f= 0.54136 Hz) (A= 0.0036334) SNR=41.086

Tabla 4.10. Análisis de resultados obtenidos con el algoritmo de Prony y el programa PSAT prueba con la variable voltaje en terminales.

Eigenvalores calculados PSAT	Eigenvalores calculados con el algoritmo de Prony			
	Nodo 1	Nodo 2	Nodo 3	Nodo 4
	Voltaje (orden 146)	Voltaje (orden 138)	Voltaje (orden 137)	Voltaje (orden 153)
-0.5554± j6.594 (f = 1.0495 Hz)	**-0.12942±3.4186i** (f = 0.54409 Hz) (A=0.0026569)	**-0.12956±3.4186i** (f = 0.54408 Hz) (A=0.0033476)	**-0.13016±3.4184i** (f 0.54406Hz) (A= 0.0010777)	**-0.41069±6.7177i** (f = 1.0691Hz) (A= 8.9168e-005)
-0.5664± j6.785 (f = 1.0799Hz)	**-0.4384±6.815i** (f = 1.0846 Hz) (A=4.8901e-005)	**-0.31218±6.9136i** (f = 1.1003Hz) (A=2.9727e-005)	**-0.36875±6.646i** (f= 1.0577Hz) (A= 5.9463e-005)	**-0.12958±3.418i** (f 0.54399Hz) (A= 0.0014576)
-0.1307± j3.413 (f 0.54329Hz)	**-1.4793±14.788i** (f = 2.3536Hz) (A=8.0266e-007)	**-1.2602±14.211i** (f = 2.2617Hz) (A=4.2259e-007)	**-1.0846±15.671i** (f = 2.4942Hz) (A= 1.3071e-006)	**-0.86568±12.994i** (f = 2.068 Hz) (A= 1.1801e-006)
	-1.3584±18.297i (f = 2.912Hz) (A=1.5649e-006)	**-1.0383±20.075i** (f = 3.195Hz) (A=1.1735e-006)	**-1.8446±20.13i** (f =3.2037Hz) (A= 2.7988e-006)	**-1.981±17.434i** (f= 2.7747Hz) (A= 2.1324e-006)
	SNR=41.978	SNR=40.92	SNR=43.718	SNR=40.038

4.2.3 Análisis de la prueba

Al realizar las pruebas con el sistema de potencia simulado se destaca que es importante tomar en cuenta la amplitud como parámetro para desechar valores espurios en los resultados obtenidos como eigenvalores oscilatorios en el rango de las oscilaciones de baja frecuencia, para la tabla 4.11 se toman en cuenta sólo los eigenvalores que presenten amplitudes mayores a 0.001, realizando un resumen de los eigenvalores más cercanos a la simulación en PSAT de las tablas 4.9 y 4.10, así es posible tener en cuenta los valores más representativos del método de Prony.

En la tabla 4.11 se agregan los eigenvalores obtenidos con paqueterías comerciales con el propósito de validar los resultados del método, analizando cada variable por separado; mostrando así que la potencia activa es la variable con la cual se obtienen mejores resultados en la simulación computacional en el tiempo, al utilizar menor orden para detectar oscilaciones y presentar con mayor precisión los resultados esperados para la simulación del sistema de potencia de dos áreas.

Tabla 4.11. Comparación de los eigenvalores de oscilaciones de baja frecuencia utilizando el método de Prony con algunas paqueterías comerciales (Adaptado de [6]).

PSS/E	DigSILENT	EUROSTAG	Kundur [14]	SSAT	PSAT	Prony
-0.613 ± j6.76 (f = 1.075 Hz) -0.631 ± j6.94 (f = 1.105 Hz) -0.155 ± j3.41 (f = 0.543 Hz) +0.187 ± j0.163 (f = 0.026 Hz) -0.149 ± j0.067 (f = 0.011 Hz)	-0.626 ± j6.67 (f = 1.062 Hz) -0.636 ± j6.90 (f = 1.097 Hz) -0.140 ± j3.41 (f = 0.542 Hz)	-0.558 ± j6.58 (f = 1.047 Hz) -0.570 ± j6.77 (f = 1.077 Hz) -0.115 ± j3.42 (f = 0.544 Hz) -0.0998 ± j0.030 (f = 0.005 Hz)	-0.492 ± j6.82 (f = 1.087 Hz) -0.506 ± j7.02 (f = 1.117 Hz) -0.111 ± j3.43 (f = 0.545 Hz) -37.89 ± j0.142 (f = 0.023 Hz) -38.01 ± j0.038 (f = 0.006 Hz) -0.00076 ± j0.0022 (f = 0.0003 Hz)	-0.5125 ± j7.2365 (f = 1.1197 Hz) -0.5145 ± j7.2365 (f =1.1517 Hz) -0.1022 ± j4.3849 (f = 0.6979 Hz)	-0.5554± j6.594 (f = 1.0495 Hz) -0.5664± j6.785 (f = 1.0799Hz) -0.1307± j3.413 (f = 0.5432Hz)	**POTENCIA ACTIVA** **-0.43096 ± j6.4999** (f = 1.0345 Hz) (A=0.0044396) BUS 3 **-0.67681 ± j6.7922** (f = 1.081Hz) (A=0.0074131) BUS2 **-0.12979 ± j3.4176i** (f= 0.54393 Hz) (A= 0.06913) BUS4 **POTENCIA REACTIVA** **-0.49566 ± j6.6406** (f = 1.0569 Hz) (A= 0.0022558) BUS 4 **-0.12942 ± j3.4186i** (f = 0.54409Hz) (A=0.0026569) BUS3 **VOLTAJE** **-0.12958 ± j3.418i** (f 0.54399Hz) (A= 0.0014576) BUS4

4.3 Implementación utilizando simulador de sistemas de potencia en tiempo real

4.3.1 Descripción de la Prueba

Se realiza la prueba en línea utilizando el simulador digital en tiempo real cargando el sistema de potencia de dos áreas, la información del proceso de cargado de datos y aspectos generales acerca del simulador se encuentra en [31].

Se miden el voltaje y corriente instantáneos directamente del simulador, estas variables son representadas con señales de voltaje en el rango de 0.5 a 2.5 volts.

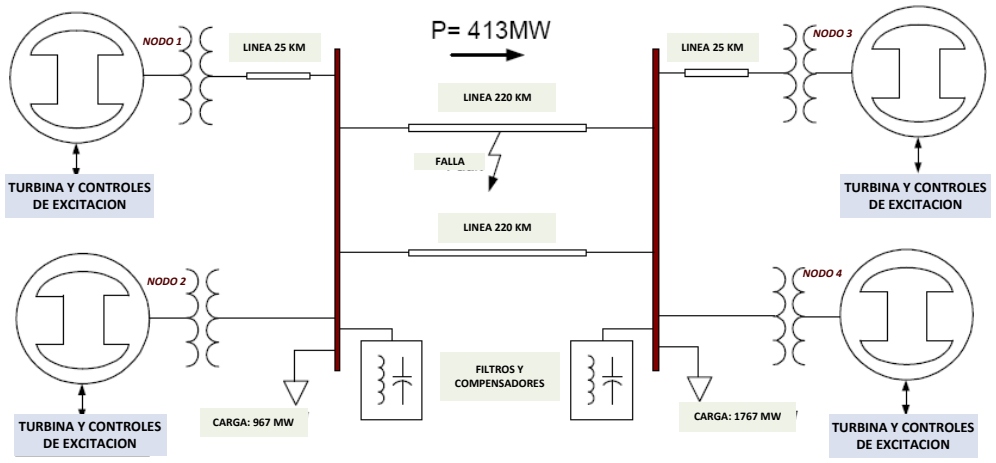


Figura 4.10. Sistema de potencia de dos áreas utilizado del simulador de sistemas de potencia en tiempo real.

Una vez cargado el sistema de dos áreas en el simulador se obtienen las mediciones utilizando los canales del ADC del microcontrolador, que mediante un cable (USB–RS232) se encarga de enviar la variable (magnitud de voltaje, potencia activa o potencia reactiva) al programa de Prony que se encuentra programado en la computadora portátil.

En el programa de Prony se realizan pruebas moviendo el orden del polinomio aproximación hasta llegar a valores de SNR mayores a 40 db, la prueba se realiza analizando cada nodo por separado y del mismo modo cada variable.

La prueba es realizada en línea enviando una variable cada 0.03 segundos, el programa de Prony se encarga de acumular el vector de mediciones hasta obtener un periodo de tiempo de 10 segundos con 300 variables, en los resultados mostrados se observan los eigenvalores oscilatorios que se asemejan más a las simulaciones realizadas previamente.

4.3.2 Resultados de la Prueba

Tabla 4.12. Resultados obtenidos de la implementación del algoritmo de Prony y el programa PSAT prueba utilizando la magnitud de voltaje.

Magnitud de voltaje			
	(con t de muestreo de 0.0333)	(con t de muestreo de 0.1)	Eigenvalores calculados PSAT
Nodo 1	-0.63045 ± 6.5105i (f= 1.0362 Hz) -0.17521 ± 3.4033i (f=0.54165 Hz) -0.2616 ± 6.6813i (f=1.0634 Hz)	-0.5757 ± 6.5766i (f=1.0467 Hz) -0.12409 ± 3.5857i (f=0.57069 Hz)	
Nodo 2	-0.56048 ± 6.6394i (f= 1.0567Hz) -1.612 ± 3.1605i (f=0.50302 Hz) -0.52901 ± 6.7029i (f=1.0668 Hz)	-0.37549 ± 6.5865i (f=1.0483 Hz) -0.1984 ± 3.279i (0.52187) -0.52041 ± 6.8813i (f=1.0952 Hz)	-0.5554± j6.594 (f = 1.0495 Hz) -0.5664± j6.785 (f = 1.0799Hz)
Nodo 3	-0.53063 ± 6.5915i (f= 1.0491 Hz) 0.29192 ± 3.4642i (f= 0.55135 Hz)	-0.1905 ± 3.5235i (f= 0.56079 Hz) -0.63808 ± 6.7397i (f= 1.0727 Hz)	-0.1307± j3.413 (f 0.54329Hz)
Nodo 4	--0.68263 ± 6.5771i (f= 1.0468 Hz) 0.15268 ± 3.4074i (f= 0.54231 Hz) -0.53127 ± 6.7123i (f= 1.0683 Hz)	-0.56113 ± 6.4627i (f= 1.0286 Hz) -0.94275 ± 3.5512i (f= 0.56519 Hz) -0.87569 ± 6.7861i (f= 1.08 Hz)	

Tabla 4.13. Resultados obtenidos de la implementación del algoritmo de Prony y el programa PSAT prueba utilizando Potencia Activa.

Potencia Activa			
	(con t de muestreo de 0.0333)	(con t de muestreo de 0.1)	Eigenvalores calculados PSAT
Nodo 1	-0.51383 ± 6.4626i (f= 1.0286 Hz) -0.10026 ± 3.6588i (f=0.58232 Hz) -0.53947 ± 6.6714i (f=1.0618Hz)	-0.22129 ± 6.6489i (f=1.0582Hz) -0.33498 ± 3.4658i (f=0.5516Hz) -0.36252 ± 6.7912i (f=1.0809 Hz)	-0.5554± j6.594 (f = 1.0495 Hz) -0.5664± j6.785 (f = 1.0799Hz) -0.1307± j3.413 (f 0.54329Hz)
Nodo 2	-0.53936 ± 6.4382i (f= 1.0247 Hz) -0.50566 ± 3.4218i (f=0.5446Hz) -0.52901 ± 6.7029i (f=1.0668 Hz)	-0.068539 ± 6.5781i (f=1.0469 Hz) -0.33498 ± 3.4658i (f= 0.5516 Hz) -0.52354 ± 6.7094i (f=1.0678 Hz)	
Nodo 3	-0.66717 ± 6.5749i (f= 1.0464 Hz) -1.2201 ± 3.0571i (f= 0.48655 Hz) -0.63691 ± 6.6285i (f= 1.055 Hz)	-0.22015 ± 6.5475i (f= 1.0421Hz) -0.58044 ± 3.0979i (f= 0.49305 Hz)	
Nodo 4	-0.72005 ± 6.4959i (f= 1.0339 Hz) -0.16593 ± 3.4587i (f= 0.55046 Hz) -0.28982 ± 6.846i (f= 1.0896 Hz)	-0.1912 ± 3.6038i (f= 0.57356 Hz) -0.45002 ± 6.979i (f= 1.1107 Hz)	

Tabla 4.14. Resultados obtenidos de la implementación del algoritmo de Prony y el programa PSAT prueba utilizando Potencia Reactiva.

Potencia Reactiva			
	(con t de muestreo de 0.03)	(con t de muestreo de 0.1)	Eigenvalores calculados PSAT
Nodo 1	-0.13535 ± 3.3267i (f=0.58232 Hz) 0.51623 ± 6.7872i (f=1.0802 Hz)	-0.19491 ± 3.402i (f=0.54144 Hz)	5554± j6.594 (f = 1.0495 Hz) -0.5664± j6.785 (f = 1.0799Hz) -0.1307± j3.413 (f 0.54329Hz)
Nodo 2	-0.47921 ± 6.6791i (f=1.063Hz) -0.17298 ± 3.1809i (f=0.50625 Hz)	-0.52066 ± 6.5939i (f=1.0494 Hz) -0.1267 ± 3.3479i (f= 0.53283 Hz)	
Nodo 3	-0.58095 ± 6.5719i (f= 1.046Hz) -0.45983 ± 3.3871i (f= 0.53907 Hz) -0.49112 ± 6.7561i (f= 1.055 Hz)	-0.74627 ± 6.5543i (f= 1.0431 Hz) -0.61739 ± 3.5261i (f= 0.56119 Hz)	
Nodo 4	-0.45698 ± 6.6182i (f= 1.0533 Hz) -0.38848 ± 3.4608i (f= 0.5508 Hz)	-0.54968 ± 6.5962i (f= 1.0498 Hz) -0.21799 ± 3.3075i (f= 0.52641 Hz)	

4.3 Análisis de la Prueba

En los resultados obtenidos de las tablas 4.12, 4.13 y 4.14 se incluyen dos tiempos de muestreo; el tiempo de muestreo de 0.03 segundos con el cual se realizó la implementación con el microcontrolador en línea y un muestreo de 0.1 utilizando las variables guardadas en la memoria USB en una prueba fuera de línea.

En la implementación en línea y fuera de ella, el método logra detectar los eigenvalores de baja frecuencia que se encontraron en el análisis del sistema de potencia en la simulación computacional, aunque con menor precisión debido al ruido presente en las señales de entrada.

CAPITULO 5

CONCLUSIONES.

5.1 Conclusiones

Como conclusión general de este trabajo se puede decir que se consigue el objetivo planteado al detectar oscilaciones de baja frecuencia a partir de mediciones de fasores utilizando un microcontrolador, esto se realizó al ejecutar la prueba final que fue validada con simulaciones previas.

5.1.1 Conclusiones del Prototipo implementado

De este trabajo se puede concluir que es posible realizar una Unidad Medidora Fasorial (PMU) utilizando el sistema operativo en tiempo real MQX, siendo necesario poner atención en la sincronización y tiempo de activación de las diferentes tareas. Estos dos puntos son los de mayor importancia dado que cualquier error en éstos, entregará como resultados datos errados en la medición de fasores.

El prototipo implementado permite guardar resultados en una memoria USB para un análisis posterior. Y cuenta con un puerto de comunicación para transmitir datos en línea a una PC para detectar oscilaciones de baja frecuencia cada 10 segundos.

5.1.2 Conclusiones del Algoritmo de Prony para la detección de oscilaciones de baja frecuencia.

En el capítulo 4 al analizar el algoritmo utilizando señales ideales se logró comprobar que es posible detectar los eigenvalores oscilatorios de una señal en el rango de 0.5 – 3.5 Hz.

En esta prueba se concluye que para utilizar este método es necesario que el orden del polinomio de aproximación tenga que ser más alto que el orden de la señal de entrada aumentando precisión en los resultados.

A medida que se aumenta el orden del método de Prony se logra con más exactitud encontrar los eigenvalores oscilatorios de la señal de entrada, pero esta medida añade eigenvalores espurios en los respuestas del método.
Un concepto muy utilizado para determinar el orden en el método de Prony, es que la relación señal ruido (SNR) debe ser es mayor o igual a 40 db, este valor se utiliza como un criterio para determinar cuando el método logró una buena aproximación de la señal de entrada.

El tiempo de muestreo utilizado cuando se trabajan con oscilaciones a frecuencias cercanas a 0.5 Hz, es crítico, al utilizar más muestras por ciclo el orden del polinomio tiende a ser mayor, y por el contrario a muy pocas muestras por ciclo se altera el resultado al aparecer eigenvalores espurios de baja frecuencia que no pertenecen a la señal de entrada, como se puede observar en tabla 4.8.

Al trabajar con las oscilaciones provenientes de la simulación de un sistema de potencia utilizando el programa PSAT, se utilizan señales no ideales y que se acercan más a la realidad, el método logra detectar los eigenvalores oscilatorios que se encontraron en la literatura y las simulaciones de estabilidad a pequeños disturbios que se utilizaron, concluyendo, de la simulación que la variable más adecuada para detectar las oscilaciones, es la potencia activa, como se observa en la tabla 4.11.

La amplitud de los eigenvalores sirve para descartar los modos espurios en el rango de las oscilaciones de baja frecuencia. En la Tabla 4.11 se realiza un resumen de eigenvalores donde se muestra esta característica, al tomar en cuenta solo los eigenvalores representativos de los resultados del programa de Prony.

En las pruebas realizadas mediante el simulador en tiempo real al prototipo se observó que es posible detectar los eigenvalores utilizando mediciones provenientes de un microcontrolador.

5.2 Aportaciones

- *Documentación de la implementación hardware y software de un prototipo para que pueda ser utilizado en trabajos futuros sobre detección de oscilaciones de baja frecuencia o análisis de la estabilidad a pequeños disturbios.*

- *Programas en lenguaje c con el sistema operativo en tiempo real MQX 3.6 de las tareas para implementar en un microcontrolador la medición y comunicación de fasores de voltaje y corriente, potencia activa y reactiva.*

- *Programa en MATLAB para la detección de oscilaciones de baja frecuencia mediante el algoritmo de Prony y datos provenientes de un microcontrolador a través de un puerto de comunicación serial.*

5.3 Recomendaciones para trabajos futuros

- *En cuanto la programación del microcontrolador se puede agregar un GPS para la utilización de mas microcontroladores interviniendo en la medición de fasores en un sistema de potencia, permitiendo mediciones sincronizadas de la señales.*

- *Comunicación vía Ethernet aun centro de control para cumplir con estándares internacionales.*

- *En cuanto al método de Prony, es posible mejorar el algoritmo implementado utilizando la descomposición en valores singulares o el método de componentes principales.*

- *Probar la detección de oscilaciones de baja frecuencia mediante el método de Fourier.*

- *Implementar dentro del microcontrolador el algoritmo de detección de oscilaciones de baja frecuencia para que formara parte de un sistema embebido.*

REFERENCIAS

[1] J. F. Hauer, C.J. Demeure and L.L. Scharf. "Initial Results in Prony Analysis of Power System Response Signals". IEEE Transactions on Power Systems. Vol. 5, No. 1, pp. 80-89, February 1990.

[2] J. F. Hauer, "Application of Prony analysis to the determination of modal content and equivalent models for measured power system response". IEEE Transactions on Power Systems. Vol. 6, No. 3, pp. 1062-1068, August 1991.

[3] C. E. Grund, J. J. Paserba, J. F. Hauer and S. Nilsson. "Comparison of Prony Analysis and Eigenanalysis for Power System Control Design". IEEE Transactions on Power Systems, Vol. 8, No.3, pp 964-971, August, 1993.

[4] CIGRE Task Force 38.01.07, Analysis and Control of Power System Oscillations, Task Force 07 of Advisory Group 01 of Study Committee 38, Technical Brochure No. 111. December 1996.

[5] D. J. Trudnowski, J.M. Johnson and J.F. Hauer. "Making Prony Analysis More Accurate using Multiple Signals". IEEE Transactions on Power Systems. Vol. 14, No. 1, pp. 226-231, February 1999.

[6] K. K. Kaberere, K. A. Folly, M. Ntombela and A. I. Petroianu. "Comparative Analysis and Numerical Validation of Industrial-Grade Power System Simulation Tools: Application to Small Signal Stability". Proceedings of the Power Systems Computation Conference PSCC'2005, (Session 32, Paper 3), August 22-26, 2005, Liege, BELGIUM.

[7] L. Ding, A. Xue, F. Han, J. Maohai and W. T. B. Wang. "Dominant Mode Identification for LowFrequency Oscillations of Power Systems basedon Prony Algorithm". IEEE Transactions on Power Systems 2010.

[8] N. Zhou, Z. Huang, F. Tuffner, J. Pierre, S. Jin. "Automatic Implementation of Prony Analysis for Electromechanical Mode Identification from Phasor Measurements" IEEE Transactions on Power Systems 2010.

[9] W. Liang, H. Kang, L. Yao. "Detection of Power System Oscillation Using Moving Window Prony Method". IEEE Transactions on Power Systems 2010. International Conference on Power System Technology

[10] L. L. Scharf. "Statiscal Signal Procesing Detecction, Estimation, and Time Series Analysis". Addison-Wesley Publishing Company, 1991.

[11] P. Kundur. Power System Stability and Control. 1st Edition, The EPRI Power System Engineering Series, McGraw-Hill Inc., New York NY, U.S.A., 1994.

[12] G. Rogers. Power System Oscillations. Kluwer Academic Press, 2000.

[13] M. A. Pai, D. P. Sen Gupta and K. R. Padiyar. Small Signal Análysis of Power Systems. 1a. Edición. Alpha Science International LTD. 2005. pp 199-212.

[14] E. W. Tan "Power System Dynamic Security Assignment via Prony analysis", Tesis para obtener el grado de Bachelor en Ingenieria de University of Queensland, Australia 2003.

[15] O. Moreno Reyes. "Identificación de la Estabilidad a Pequeños Disturbios de la Máquina Síncrona Bus-Infinito por Redes Neuronales" Tesis para obtener el grado de M. C. en Ing. Eléctrica, S.E.P.I-E.S.I.M.E I.P.N., México D.F 2005.

[16] C. Cuvas Castillo. "Implementación de un medidor fasorial". Tesis para obtener el grado de M. C. en Ing. Eléctrica, S.E.P.I-E.S.I.M.E I.P.N., México D.F 2006.

[17] D. Villarreal Martínez. "Análisis modal de sistemas eléctricos de potencia". Tesis para obtener el grado de M. C. en Ing. Eléctrica, S.E.P.I-E.S.I.M.E I.P.N., México D.F 2008.

[18] B. V. Hernández Gómez. "Diseño e implementación de un medidor fasorial síncrono normalizado con el estándar IEEE C37.118". Tesis para obtener el grado de M. C. en Ing. Eléctrica, S.E.P.I-E.S.I.M.E I.P.N., México D.F 2009.

[19] G. R. B. Prony. "Essai Experimental el Analyti". J. de L'École Polytechnique (Paris); Vol. EI, 1795, pp. 24-76.

[20] CodeWarrior Development Studio for ColdFire Architectures V7.2 Quick Start

[21] Freescale MQXTM, Real-Time Operating System, User's Guide, Document Number: MQXUG, Rev. 1, 01/2010.

[22] Freescale MQX™ RTOS, Reference Manual, Document Number: MQXRM, Rev. 3,01/2010.

[23] S. W. Smith "The scientist and Engineer s Guide to digital Signal Procesing " Second edition

[24] J. G. Slootweg, J. Persson, A. M. Van Voorden, G. C. Paap and W. L. Kling. "A Study of the Eigenvalue Analysis Capabilities of Power System Dynamics Simulation Software". Proceedings of the Power Systems Computation Conference PSCC'2002. (Session 26, Paper 3). Junio 2002.

[25] PTI, PSS/ETM 29 Program Application Guide, October 2002

[26] DIgSILENT Manuals, DIgSILENT PowerFactory, Version 13, 2003

[27] EUROSTAG Release 4.2 Package Documentation Part I, Octubre 2002

[28] Power System Analysis Toolbox (PSAT)
http://www3.uclm.es/profesorado/federico.milano/software.htm

[29] Dynamic Security Assessment Software (DSA TOOLS).
www.dsatools.com/

[30] M. A. Alvarez, G. Rosas. "Manual operativo del simulador digital en tiempo real de OPAL-RT Technologies- Revisión 1". Reporte interno SEPI-IE12_01, Enero 2012.

[31] Freescale MCF5225x Family, "Fact sheet" Document number: KRN3MCF5225XFS/ REV 2, 2008.

[32] Freescale M52259DEMOCOM, "Hardware User Guide", Document Number: 0453-010, Rev. A, 10/2008

[33] Freescale MQX™, USB Host User's Guide, Document Number: MQXUSBHOSTUG, Rev. 1, 1/2009.

BIBLIOGRAFÍA

J. Sanchez Gasca and J.H. Chow, "Performance Comparison of Three Identifications Methods for the Analysis of Electromechanical Oscillations". IEEE Transactions on Power Systems, Vol. 14, No. 3, pp. 995-1002, August 1999.

R. Kumaresan, D. W. Tufts, L. L.Scharf "A Prony Method for Noisy Data: Choosing the Signal Components and Selecting the Order in Exponential Signal Models". Proc IEEE, 1984, 72(2): 230-233.

C. A. Juárez, D.G. Colomé, (2009). "Tendencias en la supervisión en tiempo real de la estabilidad de pequeña señal de sistemas de potencia", XIII eriac décimo tercer encuentro regional iberoamericano de cigré

A. G. Phadke. "Time synchronization techniques, coupled with the computer-based measurements techniques, provide a novel opportunity to measure phasors and phase angle differences in real time". IEEE Computer Applications in Power. April 1993. 11-15

IEEE Standard for Synchrophasors: IEEE Std. 1344–1995, copyright 1996 by IEEE, 345 East 47th Street, New York, NY 10017–2394.

D. G. Hart, D. Uy, V. Gharpure, D. Novosel, D. Karlsson, M. Kaba, (2001). "Unidades PMU Supervisión de las redes eléctricas: un nuevo enfoque", Revista ABB

J. Depablos, V. Centeno, A. G. Phadke, M,Ingram, (2005),"Comparative Testing of Synchronized Phasor Measurement Units",IEEE 2005.

APÉNDICE A

DATOS SISTEMAS DE POTENCIA DE DOS ÁREAS

Este sistema es muy utilizado para el análisis de oscilaciones de baja frecuencia debido a su simetría. Este sistema consta de dos áreas similares, cada área contiene un par de unidades de generación. Existen diferentes versiones de este sistema, los datos utilizados en este trabajo se obtuvieron de [10].

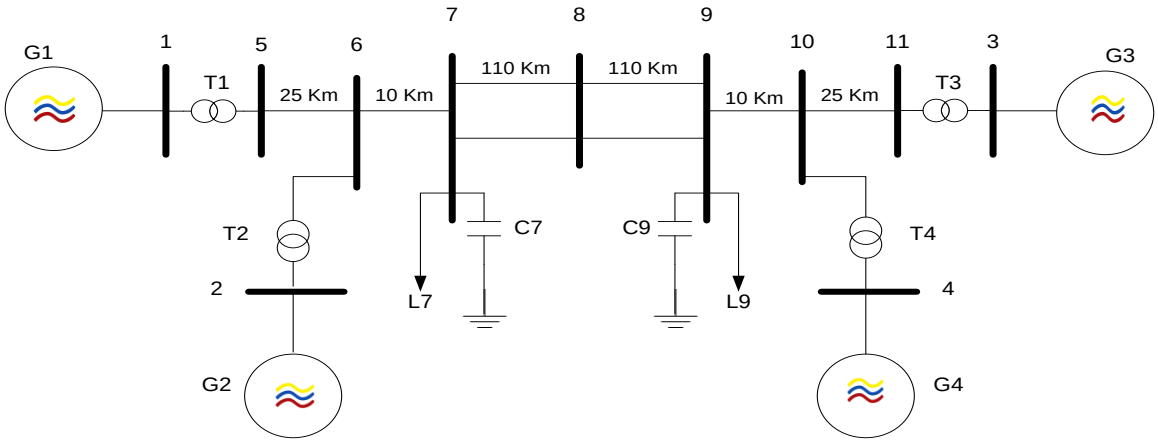


Figura A.1. Diagrama unifilar del sistema de dos áreas de [10].

Tabla A.1. Datos de la red de transmisión.

Buses terminales		Impedancia serie		Tap		No. Cto.	$B/2$	Elemento
Nodo i	Nodo j	R_i	X_i	$Mag.$	$Angulo$			
1	5	0	0.016667	1	0	1	0	Transf. 1
2	6	0	0.016667	1	0	1	0	Transf. 2
5	6	0.0025	0.025	0	0	1	0.02188	Línea 1
6	7	0.001	0.010	0	0	1	0.00875	Línea 2
7	8	0.011	0.11	0	0	1	0.09625	Línea 3
7	8	0.011	0.11	0	0	2	0.09625	Línea 4
8	9	0.011	0.11	0	0	1	0.09625	Línea 5
8	9	0.011	0.11	0	0	2	0.09625	Línea 6
9	10	0.001	0.01	0	0	1	0.00875	Línea 7
10	11	0.0025	0.025	0	0	1	0.02188	Línea 8
4	10	0	0.016667	1	0	1	0	Transf. 4
3	11	0	0.016667	1	0	1	0	Transf. 3

Tabla A.2. Datos de los buses.

nodo	V	∂	P_G	Q_G	P_d	Q_d	P_{sh}	Q_{sh}
1	1.029994	20.2003	7.01174	1.894401	0	0	0	0
2	1.010003	10.3962	7	2.446621	0	0	0	0
3	1.029996	-7.2034	7.19	1.930478	0	0	0	0
4	1.010001	-17.4413	7	2.431114	0	0	0	0
5	1.005760	13.7230	0	0	0	0	0	0
6	0.976486	3.6026	0	0	0	0	0	0
7	0.959300	-4.8457	0	0	9.670	1	0	2
8	0.940315	-18.8831	0	0	0	0	0	0
9	0.959300	-32.7801	0	0	17.670	1	0	3.5
10	0.976737	-24.2331	0	0	0	0	0	0
11	1.005512	-13.8477	0	0	0	0	0	0

Tabla A.3. Parámetros de la maquina síncrona del sistema.

Parámetro	Nodo al que está conectada la maquina			
	1	2	3	4
H	6.5	6.5	6.175	6.175
D	0	0	0	0
R_a	0.0025	0.0025	0.0025	0.0025
X_l	0.2	0.2	0.2	0.2
X_d	1.8	1.8	1.8	1.8
X_q	1.7	1.7	1.7	1.7
$X_d^{'}$	0.3	0.3	0.3	0.3
$X_q^{'}$	0.55	0.55	0.55	0.55
$T_{d0}^{'}$	8	8	8	8
$T_{q0}^{'}$	0.4	0.4	0.4	0.4
$X_d^{''}$	0.25	0.25	0.25	0.25
$X_q^{''}$	0.25	0.25	0.25	0.25
$T_{d0}^{''}$	0.03	0.03	0.03	0.03
$T_{q0}^{''}$	0.05	0.05	0.05	0.05
A_{sat}	0.015	0.015	0.015	0.015
B_{sat}	9.6	9.6	9.6	9.6
Ψ_{TI}	0.9	0.9	0.9	0.9

Tabla A.4.Parámetros para el sistema de excitación tipo DC1.

Parámetro	Nodo al que está conectada la maquina			
	1	2	3	4
K_A	0	5	6	5
T_A	0	0.060	0.05	0.06
T_B	0	0	0	0
T_C	0	0	0	0
K_E	0	-0.05	-0.06	-0.02
T_E	0	0.25	0.41	0.5
K_F	0	0.040	0.057	0.08
T_F	0	1	0.5	1
S_{E1}	0	0.08	0.66	0.13
S_{E2}	0	0.26	0.88	0.34
V_{min}	0	-1	-1	-1
V_{max}	0	1	1	1

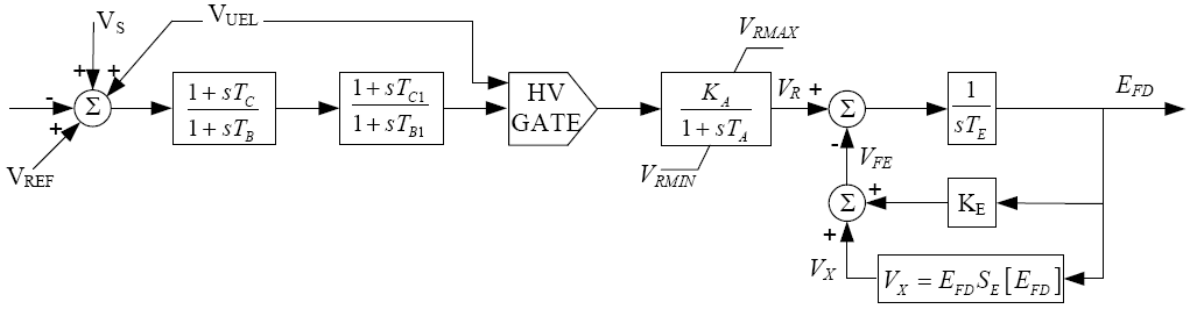


Figura A.2. Sistema de excitación DC1 [17].

APÉNDICE B

HARDWARE

B.1 Introducción

En este apartado se presenta la descripcion del hardware utilizado en la implementación del proyecto, desde la medicion de fasores hasta la prueba final utilizando el simulador de sistemas de potencia, el hardware se resume en dos dispositivos básicos:

- *Microcontrolador Coldfire version 2 MCF 52259*
- *Simulador de sistemas de potencia en tiempo real descrito en [30].*

B.2 Microcontrolador Coldfire version 2 MCF 52259

Esta sección proporciona una visión general del microcontrolador utilizado mencionando sus características generales y sus componentes principales

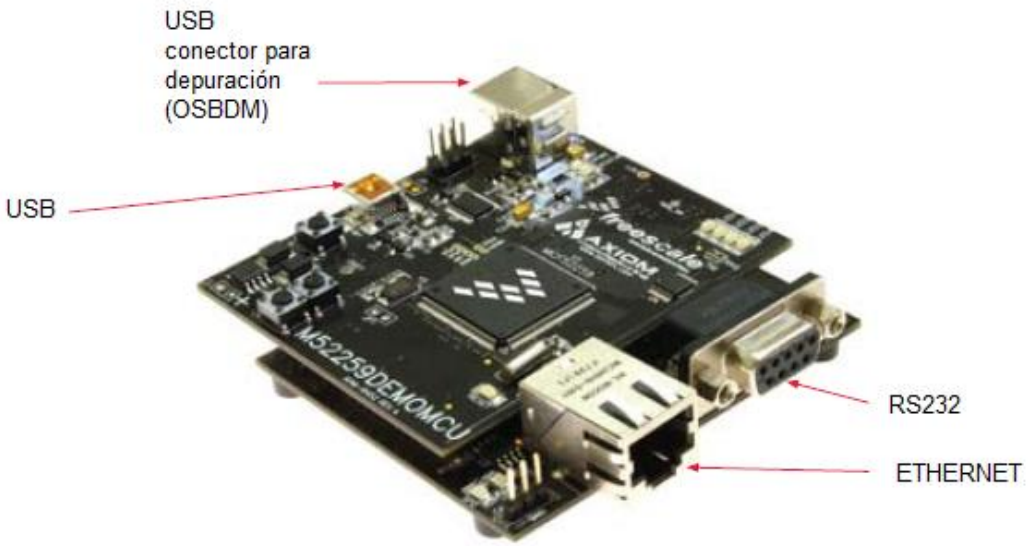


Figura B.1. Tarjeta de desarrollo MCF 52259 [31].

El MCF52259 es miembro de la familia Coldfire V2, es un dispositivo de 32 bits basado en la versión 2 con núcleo ColdFire operando a una frecuencia de hasta 80 MHz, que ofrece alto rendimiento y bajo consumo de energía. En el chip de recuerdos estrechamente conectado con el núcleo del procesador incluyen hasta 512 Kb de memoria Flash y 64 KB de memoria estática [32]. A continuación se enumeran algunos módulos incluidos en el chip:

- *Aceleración de la Unidad de cifrado (CAU)*
- *Controlador rápido de Ethernet (FEC)*
- *Bus serial universal on-the-go(USBOTG)*
- *Transceptor USB ()*
- *Modulo controlador de red de área Flex-can(CAN).*
- *Tres receptores universales asíncronos / transmisores (UARTs).*
- *Dos interfaz de bus de circuito integrado*
- *Módulo de interfaz periférica serial (QSPI).*
- *Ocho canales de 12 bits para el convertidor analógico digital (ADC) con muestreo simultáneo.*
- *Cuatro canales, 32 bits de acceso directo a memoria (DMA)*
- *Cuatro canales, 32 bits de captura de entrada / salida de temporizadores para comparar con el DMA de apoyo (DTIM).*
- *Cuatro canales con temporizador de propósito general (GPT) con captura de entrada y salida, con modulación de la amplitud de pulso (PWM), código de modulación de pulso (PCM) y acumulación de pulso.*
- *8 canales de 8 bits o 4 canales de 16 bits para el temporizador de la modulación de la amplitud e pulso.*
- *Dos temporizadores de interrupción periódica de 16 bits (PITs).*
- *Módulo de reloj de tiempo real con cristal de 32 kHz*
- *Software programable del temporizador de vigilancia.*
- *Temporizador de vigilancia secundario con reloj independiente.*
- *Controlador de interrupción capaz de manejar 57 fuentes de interrupción.*
- *Módulo de reloj con 8 MHz en el chip relajación del oscilador y lazo de fase cerrada integrada (PLL).*
- *Prueba de acceso y puerto de depuración (JTAG, BDM)*

EL chip del microprocesador cuenta con 100 pines configurados como se muestra en la figura B.2.

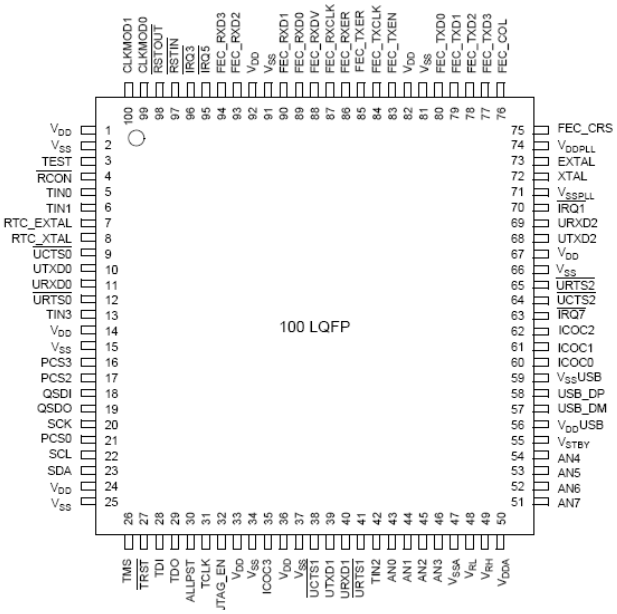


Figura B.2. Microprocesador de 100 pines [32].

B.2.1 Diagrama de bloques

El controlador embebido MCF5225x proporciona un conjunto de periféricos, la memoria de un tamaño compacto y plataforma Ethernet. En la figura B.3 adaptada de [31] se muestra el diagrama a bloques de la tarjeta de desarrollo.

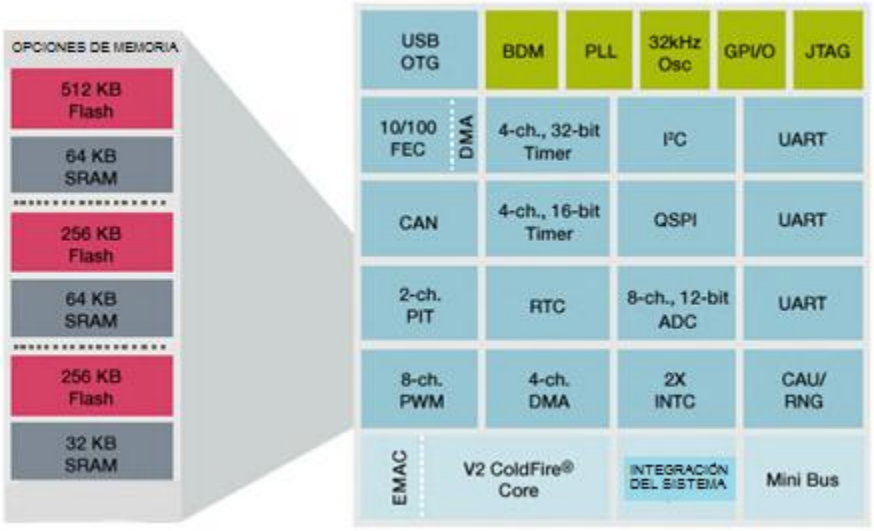


Figura B.3. Diagrama a bloques de MCF 5225x [31].

En la figura B.4 se muestra un diagrama a bloques del dispositivo más detallado con respecto al funcionamiento [32].

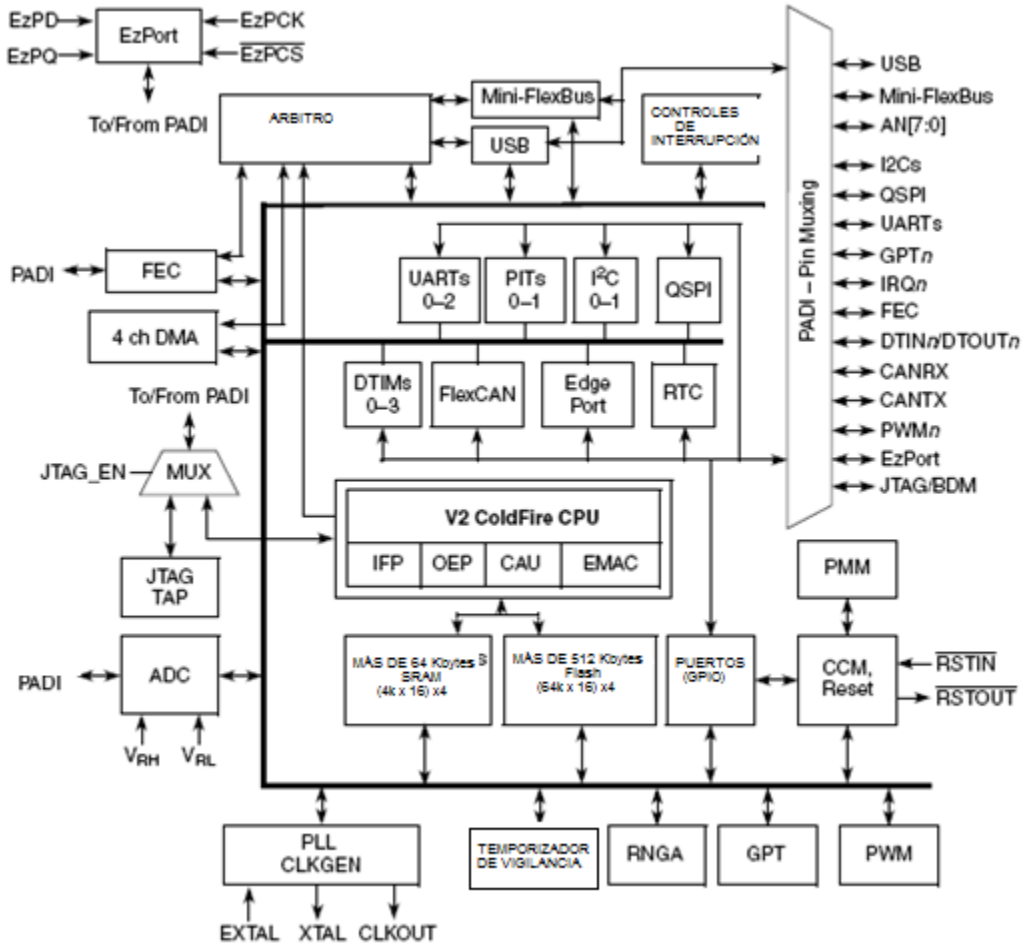


Figura B.4. Diagrama a bloques de alto nivel del dispositivo [32].

El MCF52259 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 K x 16 bits para generar 256 Kbytes de 32 bits de memoria flash. Estos conjuntos son eléctricamente borrables y programables, no volátil de datos de programa y memoria. La memoria flash es ideal para el programa y almacenamiento de datos que permite la reprogramación. La memoria flash también puede ser programada a través EzPort, que es una interfaz serie flash de programación que permite que la memoria pueda ser leída, borrada y programada por un controlador externo en un formato compatible con la mayoría del 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 [32].

B.2.2 ADC

El ADC consta de ocho canales de entrada con una resolución de 12 bits. Este almacena los resultados en un buffer accesible para su posterior procesamiento. Puede ser configurado para realizar una única medición y detenerse, realizar un análisis cada vez que es activado, o hacer 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 canales. Se pueden generar Interrupciones opcionales al final de la secuencia de exploración si un canal está fuera del rango (por debajo de las medidas de bajo umbral o limite por encima del umbral de alto límite establecido en el límite de registros) o en diferentes condiciones de cruce por cero. Entre las principales características del ADC revisadas de [32]:

- *Resolución de 12 bits*
- *Frecuencia de reloj ADC máxima de 5 MHz con periodo de 200 ns.*
- *Rango de muestreo arriba de 1.66 millones de muestras por segundo.*
- *Tiempo de conversión sencilla de 8.5 ciclos de reloj ADC 1.7 μs*
- *Tiempo de conversión adicional 6 ciclos de reloj del ADC 5.3 μs 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 secuencia y almacenamiento de ocho mediciones.*
- *Multiplexado interno para seleccionar dos de ocho entradas.*
- *Interrupción mediante eventos al finalizar de escanear, si un límite de fuera de rango es excedido.*

Los puertos del ADC utilizados para las mediciones en este proyecto son los resaltados en la figura B.5 adaptado de [33] que muestra en el puerto 29 la obtención de las muestras provenientes de la señal de voltaje y en el puerto 31 las señales medidas de corriente.

	J1		
+3.3V	1	2	RSTIN*
GND	3	4	RESET*
ICOC0	5	6	ICOC1
ICOC2	7	8	ICOC3
UTXD2/CANTX	9	10	URTS2*/I2C_SDA1
URXD2/CANRX	11	12	UCTS2*/I2C_SCL1
TIN1	13	14	TIN0
TIN3	15	16	TIN2
IRQ1*	17	18	IRQ7
QSDO	19	20	PCS3
	21	22	PCS2
QSDI	23	24	PCS0
SCK	25	26	SCL
SDA	27	28	
AN0	29	30	AN1
AN2	31	32	AN3
AN6	33	34	AN7
AN5	35	36	AN4
GND	37	38	GND
+3.3V	39	40	+3.3V

Figura B.5. Configuración de puertos del conector J1 [33].

B.2.3 Temporizadores

Existen cuatro temporizadores independientes capaces de transferir DMA de 32 bits (DTIM0, DTIM1, DTIM2 y DTIM3) en la tarjeta de desarrollo. Cada módulo incorpora un temporizador de 32 bits con un registro separado establecido para la configuración y control [31].

Los temporizadores se pueden configurar para operar desde el reloj del sistema o de una fuente externa de reloj de una de las señales, puede ser dividido por 16 o 1, este es el pre escalador. Los eventos del temporizador, opcionalmente pueden causar solicitudes de interrupción o transferencia DMA.

B.2.4 Pines de entrada y salida de propósito general (GPIO)

Casi todos los pines en el dispositivo son de propósito general de E / S y se agrupan en los puertos de 8 bits. Algunos de los puertos no utilizan todos los ocho bits. Cada puerto tiene registros para configurar, supervisar y controlar.

Estos pines permiten comunicarse con hardware externo, mediante la lectura y escritura de pines, y pueden ser utilizados para identificar una interrupción externa.

B.2.5 Reloj de tiempo real

El módulo de reloj en tiempo real provee al sistema de un reloj, cronometro, alarma y también es capaz de interrumpir rutinas. Incluye funciones completas de reloj: segundos, minutos, horas y días, es compatible con una gran variedad de funciones de interrupción de tiempo [32].

B.2.6 Controlador USB On-The-Go

Este dispositivo incluye un bus serie universal On-The-Go (USB OTG) de modo dual del controlador. USB es un estándar para la conexión de periféricos y dispositivos portátiles de electrónica de consumo tales como cámaras digitales y computadoras, entre otros.

El suplemento OTG con la especificación USB se extiende a la aplicación par por par, lo que permite a los dispositivos conectarse directamente sin la necesidad de una computadora. El controlador de modo dual en el dispositivo puede actuar como un anfitrión y USB OTG como un dispositivo USB. También es compatible con los modos de alta velocidad y baja velocidad.

B.2.7 Interfaz de comunicación serial

La interfaz de comunicación serial permite enviar y recibir información entre el microcontrolador y un dispositivo remoto, incluyendo otros microcontroladores. 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.

Se utilizó el puerto UART1, que utiliza para la intercomunicación 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 [18].

APÉNDICE C

LA PROGRAMACIÓN EN TIEMPO REAL

C.1 Introducción

En este proyecto se utilizó la programación en tiempo real que se ha convertido en un tipo de lenguaje muy utilizado en aplicaciones con sistemas embebidos, este tipo de programación surge con la exigencia a sistemas que cumplan con la ejecución en sus respuestas bajo ciertas restricciones de tiempo.

Una de las dificultades al utilizar la programación en tiempo real es la existencia de procesos concurrentes que implican un problema para la sincronización de los mismos, otro aspecto importante es que los sistemas que utilizan la programación en tiempo real trabajan a una velocidad predeterminada, lo que impone limitaciones a las tardanzas propias de un sistema real.

Para el caso de este proyecto se utiliza el sistema operativo en tiempo real MQX en su versión 3.6 de la marca Freescale, este sistema operativo se describirá a continuación.

C.2 MQX

C.2.1 Organización de MQX

MQX está constituido de los componentes del núcleo (no- opcional) y los componentes opcionales, en el caso de los componentes del núcleo, solo e muestran aquellas funciones estas funciones que MQX o la aplicación llama, para coincidir con los requerimientos se extiende el uso y la configuración de los componentes principales mediante los componentes opcionales [22].

En la figura C.1 se muestran los componentes del núcleo en el centro y los componentes opcionales alrededor de estos.

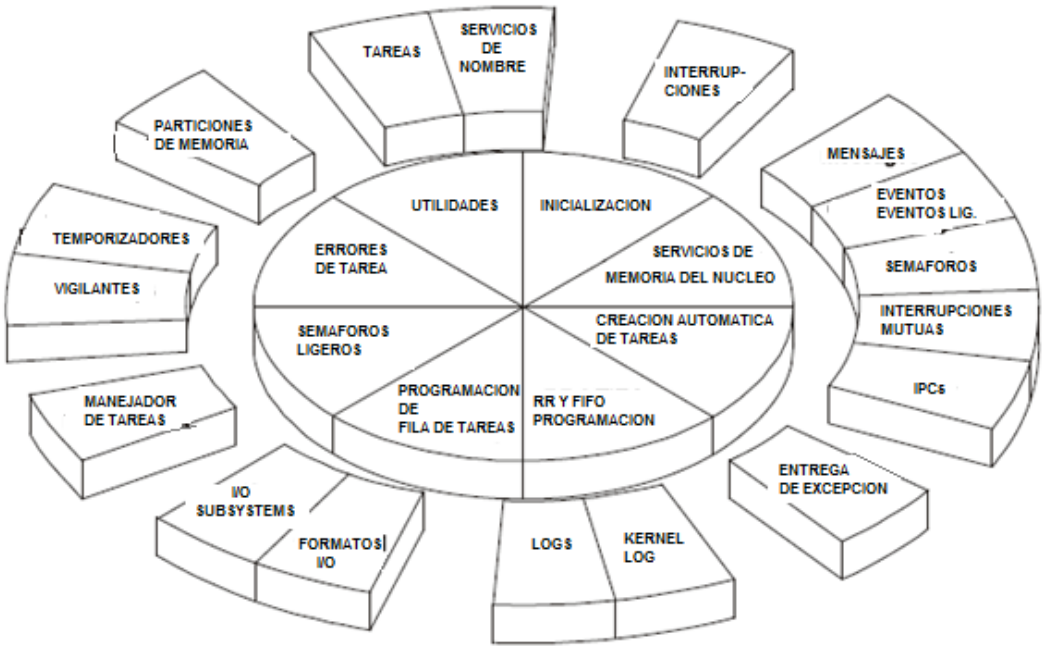


Figura C.1. Organización de MQX [22].

C.2.2 Inicialización

La inicialización es un componente del núcleo, esta aplicación comienza cuando se usa la instrucción _mqx(), cuando MQX arranca crea tareas que la aplicación define como de autoarranque [21].

C.2.3 Administrador de tareas

El administrador de tareas es parte de los componentes del núcleo.

Así como se crean automáticamente las tareas cuando MQX se inicia, una aplicación también puede crear, administrar y terminar tareas para la ejecución de la aplicación. Se pueden crear varias instancias de la misma tarea, y no hay límite para el número total de tareas en una aplicación. La aplicación puede cambiar dinámicamente los atributos de cualquier tarea. MQX libera los recursos de la tarea, cuando se termina su utilización [22].

Además, para cada tarea se puede especificar:

- *Una función de salida, que MQX llamará una vez finalizada la tarea.*
- *Un manejador de excepciones, que MQX llamará si se produce alguna excepción, mientras la tarea está activa.*

C.2.4 Programación de tareas

Para la programación de las tareas MQX incluye las siguientes opciones [21]:

- *FIFO(también llamado con base en prioridad preventiva) es componente del núcleo. Se ejecuta primero la tarea con mayor prioridad que lleva más tiempo preparada para ejecutarse.*

- *Round Robin (también llamado de rebanadas de tiempo) es un componente del núcleo. Se ejecuta la tarea de acuerdo a la tarea con mayor prioridad que lleva más tiempo preparada y que no ha consumido su porción de tiempo.*

- *Programación explicita (fila de tareas) es un componente opcional.- se puede utilizar la fila de tareas para programar explícitamente las tareas o crear un mecanismo de sincronización más complejo. Debido a que las filas de tareas proporcionan una funcionalidad mínima, son más rápidas, Un aplicación puede especificar un FIFO o Round Robin cuando se crea la fila de tareas.*

C.2.5 Sincronización de tareas

C.2.5.1 Eventos ligeros

Los eventos ligeros son componentes opcionales. Son una forma de baja sobrecarga para tareas a sincronizar utilizando pequeños cambios de estado. Los eventos ligeros requieren una cantidad mínima de memoria y se ejecutan rápidamente.

C.2.5.2 Eventos

Los eventos son un componente opcional. Estos apoyan la gestión dinámica de objetos que están formados como un campo de bits. Tareas y rutinas con servicio de interrupción pueden usar eventos para sincronizar y transmitir información simple en la forma de pequeños cambios de estado.

Los grupos de eventos pueden tener bits de auto limpieza del evento, donde MQX limpia inmediatamente el evento después de que este ha sido puesto, una aplicación puede poner un evento en un grupo de eventos que se encuentran en procesador remoto.

C.2.5.3 Semáforos ligeros

Los semáforos ligeros son componentes del núcleo. Son una forma de baja sobrecarga para tareas que comparten recursos de acceso. Los semáforos ligeros requieren un mínimo de memoria y se ejecutan rápidamente. Los semáforos ligeros cuentan con semáforos FIFO sin la herencia de prioridad.

C.2.5.4 Semáforos

Los semáforos son componentes opcionales. Se usan los semáforos para sincronizar tareas. Se puede usar un semáforo para proteger el acceso a recursos compartidos, o para mecanismo de señalización productor / consumidor. Los semáforos proporcionan filas FIFO, filas de prioridad y herencia de prioridad. Los semáforos pueden ser estrictos o no estrictos.

C.2.5.5 Exclusiones mutuas

Las exclusiones mutuas son un componente opcional. Proporciona exclusiones mutuas entre tareas, cuando se tiene acceso a recursos compartidos. Las exclusiones mutuas proveen votación, colas FIFO, filas de prioridad, herencia de prioridad y una protección prioritaria. Las exclusiones mutuas son estrictas, es decir, una terea no puede abrir una exclusión mutua, a menos que se haya cerrado primero la otra exclusión.

C.2.5.6 Mensajes

Los mensajes son un componente opcional. Las tareas pueden comunicarse con otras enviando mensajes a la fila de mensaje que se abren con otras tareas. Cada tarea abre su propia fila de entrada de mensajes. Un mensaje es identificado en la fila de mensajes mediante su identificador, que MQX asigna cuando se crea la fila. Solo la tarea que abre la cola de mensajes puede recibirlos. Cualquier tarea puede enviar un mensaje a cualquier fila previamente abierta, si conoce el identificador de la fila de tareas.

C.2.5.7 Fila de tareas

Además de proporcionar un mecanismo de programación, las filas de tareas proporcionan una forma simple y eficiente de sincronizar las tareas. Se puede suspender las tareas removiéndolas de la fila de tareas.

C.2.6 Elementos temporizadores.

C.2.6.1 Introducción

El tiempo es un componente opcional que se puede habilitar o deshabilitar a nivel de BSP. Existe el tiempo transcurrido y el tiempo absoluto, este último es posible cambiarlo. El tiempo de resolución depende de la resolución definida por la aplicación que se establece para el hardware de destino cuando se inicia MQX.

C.2.6.2 Temporizadores ligeros

Los temporizadores ligeros son un componente opcional y proporcionan un mecanismo de baja sobrecarga para el llamado de funciones de la aplicación en intervalos periódicos. Los temporizadores ligeros están instalados mediante la creación de una fila periódica, entonces agregando un temporizador para expirar algún desplazamiento desde el inicio del periodo.

C.2.6.3 Temporizadores

Los temporizadores son un componente opcional. Estos proporcionan la ejecución periódica de una función en la aplicación. MQX considera temporizadores de un solo tiro, que se ejecutan una solo vez, y temporizadores periódicos, que expiran varias ocasiones en un intervalo dado. Se pueden utilizar los temporizadores para comenzar a una hora específica o después de un tiempo determinado.

Cuando se establece un temporizador, se especifica la función de notificación que la tarea temporizador llama cuando se agota el tiempo. La función de notificación se puede utilizar para sincronizar las tareas mediante el envío de mensajes, manifestación de eventos o el uso de uno de los mecanismos de sincronización de MQX.

C.2.7 Plantillas de trabajo para tareas

La lista de plantillas de trabajo define un conjunto inicial de plantillas de las tareas que se pueden crear en el procesador.

En la inicialización, MQX crea una instancia de cada tarea, cuya plantilla la define como una tarea de inicio automático. Además, mientras se ejecuta una aplicación, puede crear otras tareas con una plantilla de tarea que se encuentra previamente definida o bien la aplicación la define dinámicamente. El final de la lista de plantillas es una tarea sin contenido. La estructura que define la plantilla de tareas:

```
typedef struct task_template_struct
{
_mqx_uint                    ÍNDICE DE LA PLANTILLA
void  _CODE_PTR_             DIRECCIÓN DE LA TAREA
_mem_size                    TAMAÑO DEL APILADO DE LA TAREA
_mqx_uint                    PRIORIDAD DE LA TAREA
char_PTR                     NOMBRE DE LA TAREA
_mqx_uint                    ATRIBUTOS DE LA TAREA
uint_32                      CREACIÓN DE PARAMETROS
_mqx_uint                    REBANADA DE TIEMPO POR DEFECTO
} TASK_TEMPLATE_STRUCT, _PTR_ TASK_TEMPLATE_STRUCT_PTR;
```

C.2.8 Asignación de la prioridad de tareas.

Cuando se asigna la prioridad de las tareas en la plantilla de tareas:

- *MQX crea una fila de tareas activa por para cada prioridad desde la más alta hasta la menor prioridad (el número más alto).*

- *Mientras se ejecuta un**a** aplicación, no se puede crear una tarea que tenga menor prioridad que la tarea que ocupa la menor prioridad en la lista de platillas.*

APÉNDICE D

CÓDIGO IMPLEMENTADO EN EL PROYECTO

D.1 Código del programa del microcontrolador

Archivo de declaración de las tareas.

```
#include <mqx.h>
#include <bsp.h>
#include "meter_config.h"
#include "task_template_list.h"

const TASK_TEMPLATE_STRUCT MQX_template_list[] =
{
    {
        /* Task Index       */  ADC_TASK,
        /* Function         */  adc_task,
        /* Stack Size       */  1000,
        /* Priority Level   */  9,
        /* Name             */  "ADC Task",
        /* Attributes       */  0,//MQX_AUTO_START_TASK,
        /* Creation Params  */  0,
        /* Time Slice       */  0,
    },

    {
        /* Task Index       */  CALCULA_TASK,
        /* Function         */  calcula_task,
        /* Stack Size       */  1000,
        /* Priority Level   */  10,
        /* Name             */  "Display Task",
        /* Attributes       */  0 ,
        /* Creation Params  */  0,
        /* Time Slice       */  0,
    },
      {
        /* Task Index       */  MAIN_TASK,
        /* Function         */  Main_task,
        /* Stack Size       */  3000,
        /* Priority Level   */  8,
        /* Name             */  "usb",
        /* Attributes       */   MQX_AUTO_START_TASK   ,
        /* Creation Params  */  0,
        /* Time Slice       */  0,
    },
    {
        /* Task Index       */  SINCRONIZA_TASK,
        /* Function         */  Sincroniza_task,
        /* Stack Size       */  3000,
        /* Priority Level   */  6,
        /* Name             */  "sincroniza",
        /* Attributes       */  0,//MQX_AUTO_START_TASK
        /* Creation Params  */  0,
        /* Time Slice       */  0,
    },

    { 0 }
};
```

Archivo tarea sincroniza

```
#include <mqx.h>
#include <bsp.h>
#include "meter_config.h"
#include "metering_algorithms.h"
#include "task_template_list.h"
#include "tad.h"
#include <event.h>


/* extern variables */
extern Power_vec    Power[4];
extern int band;
extern int_16 media;
extern int suma;
extern Complex fasor;
extern int cont;
int bandera_=0;
extern uint_32 *adc_td;
extern uint_32 *usb_td;

_task_id sincroniza_id;


void Sincroniza_task
   (
      uint_32 initial_data
   )
{

FILE_PTR port_file1;
pointer event_ptr;
static boolean  encendido=TRUE;

GPIO_PIN_STRUCT pins[] = {
  BSP_LED1 | GPIO_PIN_STATUS_0,
  BSP_LED2 | GPIO_PIN_STATUS_0,
  BSP_LED3 | GPIO_PIN_STATUS_0,
  BSP_LED4 | GPIO_PIN_STATUS_0,
  GPIO_LIST_END
};

GPIO_PIN_STRUCT pins1[] = {
  BSP_LED1 ,
  GPIO_LIST_END
};


  _event_create("global");
 _event_open("global", &event_ptr);
    puts("abre evento Task\n");

 sincroniza_id= _task_get_id();

   port_file1 = fopen("gpio:write", (char_ptr) &pins1);

    _task_block();
```

```
    while(1)

    {

     if (bandera_==2)
     {
    _task_ready(usb_td);
    _task_block();

     }

        else
  {
    _task_ready(adc_td);
    printf("\n sincroniza");
    _time_delay(20000);
  }
```

```
if (_event_set(event_ptr, 0x01) != MQX_OK)
{
printf("\nSet Event failed");
_mqx_exit(0);
}

if(encendido)
{
ioctl(port_file1, GPIO_IOCTL_WRITE_LOG1,&pins1);
encendido=FALSE;
}
else
{
encendido=TRUE;
ioctl(port_file1, GPIO_IOCTL_WRITE_LOG0,&pins1);
}


    }

}
```

Archivo tarea ADC

```
#include <mqx.h>
#include <bsp.h>
#include <string.h>
#include "meter_config.h"
#include "task_template_list.h"
#include "metering_algorithms.h"
#include "main.h"


int cont=0;
_task_id calcula_id;

extern Power_vec    Power[4];
uint_32 *adc_td,*calcula_td,*sincroniza_td;
extern uint_32 *usb_td;
int j,k;
extern entrada[SAMPLES_PER_PERIOD];
extern _task_id sincroniza_id;
extern int bandera_,cont_dft,C_j,C_k;

/* funciones prototipo */
extern void process_calculation(void);

static void adc_init(void);
static void adc_channel_init(void);
static void adc_start_measuring(void);


/* DEfinicion de resolucion */

const ADC_INIT_STRUCT adc_init_param = {
    ADC_RESOLUTION_12BIT,
};


#define ADC_L1_STR      "adc:"
#define ADC_vL1_STR     "adc:vL1"
#define ADC_iL1_STR     "adc:iL1"

/*  parametros del adc voltaje */

const ADC_INIT_CHANNEL_STRUCT vL1_channel_param =
{
/* physical ADC channel                 */ ADC_SOURCE_AN0,
/* runs continuously after PDB trigger  */ (ADC_CHANNEL_MEASURE_LOOP | ADC_CHANNEL_START_TRIGGERED),
/* number of samples in one run sequence */ SAMPLES_PER_PERIOD,
/* time offset from trigger point in us  */ ADC_VOLTAGE_DELAY,
/* period in us                         */ ADC_SAMPLING_PERIOD,
/* scale range of result (not used now) */ ADC_RESULT_RANGE_SCALE,
/* circular buffer size (sample count)  */ SAMPLE_BUFFER,
/* adc trigger                          */ ADC_TRIGGER_1,
/* lwevent object                       */ &meter_shared_event,
/* lwevent mask                         */ ADC_DATA_READY_MASK,
};

/*  parametros del adc voltaje */

const ADC_INIT_CHANNEL_STRUCT iL1_channel_param =
{
/* physical ADC channel                 */ ADC_SOURCE_AN2,
/* runs continuously after PDB trigger  */ (ADC_CHANNEL_MEASURE_LOOP | ADC_CHANNEL_START_TRIGGERED),
/* number of samples in one run sequence */ SAMPLES_PER_PERIOD,
/* time offset from trigger point in us  */ ADC_VOLTAGE_DELAY,
/* period in us                         */ ADC_SAMPLING_PERIOD,
/* scale range of result (not used now) */ ADC_RESULT_RANGE_SCALE,
/* circular buffer size (sample count)  */ SAMPLE_BUFFER,
/* adc trigger                          */ ADC_TRIGGER_1,
/* lwevent object                       */ NULL,
/* lwevent mask                         */ 0,
};
```

```
/* ADC punteros de archivos */
FILE_PTR  file_L1, file_vL1, file_iL1;


/* buffers */

int_16 vL1_buff[SAMPLES_PER_PERIOD];
int_16 iL1_buff[SAMPLES_PER_PERIOD];

uint_32 num_read_vL1;
uint_32 num_read_iL1;
```

```
void adc_task (uint_32 initial_data)
{
    static vuint_32 timeout = 0;
    static vuint_8  swap = 0;
     static boolean  encendido=TRUE;


 calcula_id = _task_create_blocked(0, CALCULA_TASK, 0);

    adc_init();


    if (_lwevent_create(&meter_shared_event, 0) != MQX_OK)
       {
//      puts("\nMake event failed!\n");
     //_task_block();
       }

     |

    _lwevent_set_auto_clear( &meter_shared_event, ADC_DATA_READY_MASK);

    /* inicializacion canales del adc */
    adc_channel_init();


    /* adc disoaro de mediciones*/
    adc_start_measuring();

    sincroniza_td=_task_get_td(sincroniza_id);
    adc_td = _task_get_td(_task_get_id());
    calcula_td=_task_get_td(calcula_id);
```

```
    while(1)
    {

       /* Espera por el evento listo puesto por el ADC */

 if (_lwevent_wait_ticks (&meter_shared_event, ADC_DATA_READY_MASK, TRUE, 0) == MQX_OK)

            /* Escribe los datos leidos por el ADC*/

   num_read_vL1 = read(file_vL1, vL1_buff, SAMPLES_PER_PERIOD * sizeof(vL1_buff[0]));
   num_read_iL1 = read(file_iL1, iL1_buff, SAMPLES_PER_PERIOD * sizeof(iL1_buff[0]));


        // realiza los calculos

    if (cont<300)
       {

        _task_ready(calcula_td);
        _time_delay(16);
        cont++;
        cont_dft++;

       }
      else
      {
        cont=0;
        _task_ready(usb_td);

       _task_block();
      }


    }
}
}


static void adc_init(void)
{
    _mqx_uint       error_code = 0;


    file_L1    = fopen(ADC_L1_STR, (const char*)&adc_init_param);
}


static void adc_channel_init(void)
{

    _mqx_uint       error_code = 0;


    file_vL1 = fopen(ADC_vL1_STR, (const char*)&vL1_channel_param);
    file_iL1 = fopen(ADC_iL1_STR, (const char*)&iL1_channel_param);

}



static void adc_start_measuring(void)
{

    ioctl(file_L1, IOCTL_ADC_FIRE_TRIGGER, (pointer) ADC_TRIGGER_1);
//     printf("triggered!\n");
}
```

Archivo tarea calcula

```
#include <mqx.h>
#include <bsp.h>
#include "meter_config.h"
#include "metering_algorithms.h"
#include "task_template_list.h"
#include "tad.h"

extern void process_calculation(void);


/* extern variables */
extern Power_vec    Power;
extern int band;
extern int_16 media;
extern int suma;
extern Complex fasor,voltaje,corriente;
extern int cont;
extern int_16 vL1_buff[SAMPLES_PER_PERIOD];
extern int_16 iL1_buff[SAMPLES_PER_PERIOD];

int  pot_P_buff[600];
int  pot_Q_buff[600];
int  Vrms_buff[600];
int  Irms_buff[600];
int  Prms_buff[600];
int  Vreal_buff[600];
int  Vimg_buff[600];
int  Ireal_buff[600];
int  Iimg_buff[600];



int cont_b1,cont_b2,cont_b3,cont_b4,cont_dft;
int C_j, C_k;
extern uint_32 *main_td;

void calcula_task
   (
      uint_32 initial_data
   )
{

   int j=0;

    while(1)
    {
     process_calculation();

     if (cont >=600)
     {
pot_P_buff[cont_dft]=Power.Act_Pwr;
pot_Q_buff[cont_dft]=Power.React_Pwr;

Vrms_buff[cont_dft] =Power.Pwr_fct;
Irms_buff[cont_dft] =Power.Mag;

Vreal_buff[cont_dft]=voltaje.Real;
Vimg_buff [cont_dft]=voltaje.Img;

Ireal_buff[cont_dft]=corriente.Real;
Iimg_buff [cont_dft]=corriente.Img;
     }


printf("%d \n",Power.Act_Pwr);

    _task_block();
```

Archivo mediciones

```
void process_calculation(void)
{
    // Calcula DFT señal de corriente y voltaja
   DFT((int_16 *) vL1_buff, &voltaje);
   DFT((int_16 *) iL1_buff, &corriente);


  Power_Calc2((int_16 *) vL1_buff,(int_16 *) iL1_buff,&Power);
  Power_Calc1((int_16 *) vL1_buff,(int_16 *) iL1_buff,&Power);

}


int_16 clear_offset(int_16 *values)
{
    uint_16 i;
    int_32 long_accu = 0;
    int_16 short_accu;


    for(i = 0; i < SAMPLES_PER_PERIOD; i++)
        long_accu += (uint_16) values[i];

    short_accu = long_accu / SAMPLES_PER_PERIOD;

    for(i = 0; i < SAMPLES_PER_PERIOD; i++)
    {
    entrada[i]=values[i];
    values[i] -=short_accu;
    }


    return short_accu;
}
```

Archivo de algoritmos de medición

```
uint_16 RMS_calc(int_16 *input)
{
    _mqx_int i;
    int_64 Sum = 0;

    for (i = 0; i < RMS_SAMPLES; i++) {
#if ASM_SUM == 1
        ADD64bits(&Sum, (*input) * (*input));
#else
        Sum = Sum + (*input) * (*input);
#endif
        input += (SAMPLES_PER_PERIOD / RMS_SAMPLES);
    }
suma=Sum;
    Sum = Sum / RMS_SAMPLES;
    return SquareRoot((uint_32) Sum);
}
```

```
void Power_Calc2(int_16 *V, int_16 *I,Power_vec *Out)
{
 //Voltaje RMS
    Out->Vrms = (RMS_calc(V) * VOLTAGE_CORRECTION_MUL) / VOLTAGE_CORRECTION_DIV;

 //Corriente RMS
    Out->Irms = (RMS_calc(I) * CURRENT_CORRECTION_MUL) / CURRENT_CORRECTION_DIV;


    Out->Mag =SquareRoot(((voltaje.Real) * (voltaje.Real))+((voltaje.Img) * (voltaje.Img)));

 // Potencia Activa
    Out->Act_Pwr = ((voltaje.Real) * (corriente.Real))+((voltaje.Img) * (corriente.Img));

//  Potencia Reactiva
   Out-> React_Pwr = (voltaje.Img * corriente.Real)-(voltaje.Real * corriente.Img);
}


void DFT(int_16 *input, Complex *Res)
{
    int_16 *p1;
    const int_16 *p2;
    int_64 Sum;
    _mqx_int i;

    if (Res == NULL)
        return;

    Sum = 0;
    p1 = input;
    p2 = &Cos_coef_k_1[0];
    for (i = 0; i < DFT_SAMPLES; i++)
    {
#if ASM_SUM == 1
        ADD64bits(&Sum, (*p1) * (*p2));
#else
        Sum = Sum + (*p1) * (*p2);
#endif
        p1 += (SAMPLES_PER_PERIOD / DFT_SAMPLES);
        p2 += (COS_SAMPLES / DFT_SAMPLES);
    }


    Sum = (Sum*2)/(DFT_SAMPLES*32767);
    Res->Real =Sum;

    Sum = 0;
    p1 = input;
    p2 = &Sin_coef_k_1[0]; // sine wave with amplitude 0x7FFF

    for (i = 0; i < DFT_SAMPLES; i++)
    {
#if ASM_SUM == 1                                // Assembly implementation of 64-bit sum
        ADD64bits(&Sum, (*p1) * (*p2));  // call Assembly function
#else
        Sum = Sum + (*p1) * (*p2);
#endif
        p1 += (SAMPLES_PER_PERIOD / DFT_SAMPLES);
        p2 += (SIN_SAMPLES / DFT_SAMPLES);
    }

    Sum =((Sum*2) / (DFT_SAMPLES*32767));
    Res->Img =Sum;
}
```

Archivo de la tarea USB

```
void Main_task(uint_32 initial_data)
{
    USB_STATUS status = USB_OK;
    FILE_PTR fp;
    static uint_32 DataToWrite = 0;
    uint_32 error;
    uint_32 data_written;
    char _PTR_ MyString;


    pointer event_ptr;
      // fecha
    TIME_STRUCT estructuraTiempoSeg;
    MQX_TICK_STRUCT estructuraTiempo;
    MQX_XDATE_STRUCT estructuraFecha;

    uint_32 anio=2011;
    uint_32 mes=12;
    uint_32 dia=1;
    uint_32 hora=12;
    uint_32 minuto=0;
    uint_32 segundo=0;



    estructuraFecha.YEAR    = anio;
    estructuraFecha.MONTH   = mes;
    estructuraFecha.MDAY    = dia;
    estructuraFecha.HOUR    = hora;
    estructuraFecha.MIN     = minuto;
    estructuraFecha.SEC     = segundo;


    _time_xdate_to_ticks(&estructuraFecha, &estructuraTiempo);
    _time_set_ticks (&estructuraTiempo);


   _lwsem_create(&gusb_sem,0);
   _event_open("global", &event_ptr);


   _int_disable();
   _int_install_unexpected_isr();
   _usb_host_driver_install(0, (pointer)&_bsp_usb_host_callback_table);


   status = _usb_host_init
       (HOST_CONTROLLER_NUMBER,
       MAX_FRAME_SIZE,
       &host_handle);
   if (status != USB_OK)
   {
       printf("USB Host Initialization failed. STATUS: %x\n", status);
       fflush(stdout);
       exit(1);
   }


   status = _usb_host_driver_info_register(host_handle, DriverInfoTable);
   if (status != USB_OK)
   {
       printf("Driver Registration failed. STATUS: %x\n", status);
       fflush(stdout);
       exit(1);
   }

   _int_enable();

//   printf("\nMQX USB\nWaiting for USB device to be attached...\n");
   fflush(stdout);
```

```
    MyString = (char _PTR_)_mem_alloc_system(MY_STRING_SIZE);

    while (1)
{

    _lwsem_wait(&gusb_sem);

    switch(gusb_state)
    {
        case USB_CONFIG_EVENT:
        /* Drop through into attach, same processing */
        case USB_ATTACH_EVENT:

//          printf("attached USB stick\n\r");
            status = _usb_hostdev_select_interface(gdev_handle, gintf_handle, &gusb_handle);
        break;
        case USB_INTF_EVENT:

            gusb_fs = usb_filesystem_install(&gusb_handle,"usb:","pm:","c:");


             if(gusb_fs)
             {

 while (1)
             {
             usb_td = _task_get_td(_task_get_id());


          adc_id = _task_create(0, ADC_TASK, 0);

            _task_block();


                fp = fopen("c:\\Potencia_activa.txt","a+");

                if(fp)
                {


                    for(i =0 ;i<499;i++)
                    {
                data_written = sprintf(MyString," %d \r\n",pot_P_buff[i]);

                    write(fp, MyString, data_written);
                        _time_delay(40);
                    }


                printf("\n\rFile Closed");
                fclose(fp);

                cont=0;

                }


                fp = fopen("c:\\Potencia_reactiva.txt","a+");
```

```
        if(fp)
        {


                for(i =0 ;i<499;i++)
                {
            data_written = sprintf(MyString," %d \r\n",pot_P_buff[i]);

                write(fp, MyString, data_written);
                    _time_delay(40);
                }


            printf("\n\rFile Closed");
            fclose(fp);

            cont=0;

        }


        fp = fopen("c:\\Potencia_reactiva.txt","a+");

        if(fp)
        {

            for(i =0 ;i<499;i++)
                {
            data_written = sprintf(MyString," %d \r\n",pot_Q_buff[i]);

                write(fp, MyString, data_written);
                    _time_delay(40);
                }

            fclose(fp);

            cont=0;

        }


    if(fp)
    {

        for(i =0 ;i<499;i++)
            {
        data_written = sprintf(MyString," %d \r\n",pot_Q_buff[i]);

            write(fp, MyString, data_written);
                _time_delay(40);
            }

        fclose(fp);

        cont=0;

    }

        fp = fopen("c:\\voltaje_rms.txt","a+");

    if(fp)
    {
        for(i =0 ;i<499;i++)
            {
        data_written = sprintf(MyString," %d \r\n",Vrms_buff[i]);

            write(fp, MyString, data_written);
                _time_delay(40);
            }

        fclose(fp);

        cont=0;

    }

    fp = fopen("c:\\corriente_rms.txt","a+");
```

```
if(fp)
{
    for(i =0 ;i<499;i++)
        {
    data_written = sprintf(MyString," %d \r\n",Irms_buff[i]);

        write(fp, MyString, data_written);
            _time_delay(40);
        }

    fclose(fp);

    cont=0;

}

    fp = fopen("c:\\voltaje_real.txt","a+");

if(fp)
{


    for(i =0 ;i<499;i++)
        {
    data_written = sprintf(MyString," %d \r\n",Vreal_buff[i]);

        write(fp, MyString, data_written);
            _time_delay(40);
        }

    fclose(fp);

    cont=0;

}


    fp = fopen("c:\\voltaje_img.txt","a+");

     fp = fopen("c:\\voltaje_img.txt","a+");

if(fp)
{


     for(i =0 ;i<499;i++)
         {
     data_written = sprintf(MyString," %d \r\n",Vimg_buff[i]);

         write(fp, MyString, data_written);
             _time_delay(40);
         }


     fclose(fp);

     cont=0;

}

     fp = fopen("c:\\corriente_real.txt","a+");

if(fp)
{


     for(i =0 ;i<499;i++)
         {
     data_written = sprintf(MyString," %d \r\n",Ireal_buff[i]);

         write(fp, MyString, data_written);
             _time_delay(40);
         }

//   printf("\n\rFile Closed");
     fclose(fp);

     cont=0;

}

     fp = fopen("c:\\corriente_img.txt","a+");
```

```
                        fp = fopen("c:\\corriente_img.txt","a+");

                    if(fp)
                    {

                        for(i =0 ;i<499;i++)
                            {
                        data_written = sprintf(MyString," %d \r\n",Iimg_buff[i]);

                            write(fp, MyString, data_written);
                                _time_delay(40);
                            }

                        printf("\n\rFile Closed");
                        fclose(fp);
                    }


                //  printf("\n activa  adc\n\r");

    _task_ready(adc_td);

        _task_block();
            }


            }

            break;
            case USB_DETACH_EVENT:

//              printf("Deattached USB stick\n\r");
                usb_filesystem_uninstall(gusb_fs);
            break;
            default:
            break;
        }
    };
}
```

D.2 Código del programa en MATLAB.

Interfaz RS 232

```
clear all;
close all;
clc;
signal=[];
t=[];
% cerrar puerto serial si ya está abierto


s = instrfind;
if ~isempty( s )
    fclose( s );
    delete( s );
    clear s
end

SerPIC = serial('COM4');
set(SerPIC,'BaudRate',115200);
set(SerPIC,'DataBits',8);
set(SerPIC,'Parity','none');
set(SerPIC,'StopBits',1);
set(SerPIC,'FlowControl','none');
fopen(SerPIC);
C=[];
for k=1:300

[A,count]=fscanf(SerPIC,'%d',16);
trans=A;
input_signal(k)=trans;
t(k)=0.0333*k;
C=[C;A];

end
```

Algoritmo de Prony

```
%-----------------    PRIMER PASO ---------------%
% PASO 1.- DETERMINAR LOS PARAMETROS DE PREDICCION
%           LINEAL QUE SE AJUSTAN A LOS DATOS DISPONIBLES.
%A y(k) = a(1)y(k-1) + a(2)y(k-2) + ..... + a(n)y(k-n)
SNR=0;
orden=60;
%orden
while (SNR <40)

orden =orden +2;
%tiempo de muestreo = 0.001
T = 0.001;

% Formacion de la matriz.

X=[];
m = length(input_signal) - orden;
step = orden;
for i = 1:orden
for j = 1:m
X(j,i) = input_signal(step-1 + j);
end
step = step-1;
end

b = [];
for l = 1:m
b(l,1) = input_signal(l+orden);
end
```

```
%-----------------  SEGUNDO PASO ---------------%
%ENCONTRAR LAS RAICES DEL POLINOMIO DE PREDICCION DEL PASO 1

%formacion del polinomio de coeficientes
%a ?=inv(X'* X)* X'* b
a = pinv(X)*b;

% z^n - [a(1)z^n-1 + a(2)z^n-2 + ... + a(n)] = 0
theta = [1 -a'];

% encontrar las raices del polinomio de coeficientes
raices = roots(theta);
lamda =(log(raices))/T;
Fr=imag(lamda)/(2*pi);

%%%% calculo de la frecuencia y factor de amortiguamiento
%                          utilizando las raices %%%%%

w=1;
Freq_bf=[];
EIG_LOWFREC=[];
for j = 1:1:orden



if  (Fr(j) >= 0.1) & (Fr(j) <= 3.5)

EIG_LOWFREC(w)=lamda(j);
Freq_bf(w)= Fr(j);
w=w+1;
end
end
```

```
%-----------------  TERCER PASO ---------------%
%CON LO YA OBTENIDO SE FORMA UN SEGUNDO SISTEMA DE ECUACIONES
%%%% formacion de matriz
Z=[];
for k = 1:(2*orden)-1
for j = 1:orden
Z(k,j) = raices(j)^(k-1);
end
end
x = [];
for n = 1:(2*orden)-1
x(n) = input_signal(n);
end

H = pinv(Z)* x';
phase_rad = atan(imag(H)/real(H));
for i = 1:1:orden
if phase_rad(:,i) ~ 0;
phase = phase_rad(:,i);
end
end

amplitude = sqrt(imag(H).^2 + real(H).^2);

AMP=[];
FASE=[];
e=1;
for j = 1:1:orden
if  (Fr(j) >= 0.1) & (Fr(j) <= 3.5)
AMP(e)=amplitude(j);
FASE(e)= phase(j);
e=e+1;
end
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%  SIGNAL TO NOISE RATIO(SNR)%%%%%%%%%%%%%%

error2=0;
mod=abs(Z*H)

for n = 1:orden
(input_signal(n)- mod(n))*(input_signal(n)- mod(n));
    error2= error2 + (input_signal(n) - mod(n))^2;

snr1(n)=input_signal(n)/(input_signal(n)-mod(n));
snr2(n)=(mod(n)-input_signal(n))/(input_signal(n));

end

snr1;
snr2;
SNRG =  20*log(norm(snr1,orden));%/sqrt(orden));
error2;
errordb=-20*log(error2);
SNR=SNRG/10;

end

%-----++++++++++++++++ RESULTADOS ------++++++----------%+


%-----+++++GRAFICAS++++++------%


% GENERACION DE GRAFICA DE LA SEÑAL DE ENTRADA%
%subplot(211),
plot(t,input_signal)
title('SEÑAL DE ENTRADA')
xlabel('TIEMPO')



%-----+++++VARIABLES++++++------%
lamda
SNR
error2;
orden
Fr;
amplitude;
AMP
FASE;
phase;
raices;
EIG_LOWFREC'
Freq_bf
```