

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

RED DE SENSORES INALÁMBRICOS PARA MICRORREDES ELÉCTRICAS

Brayan Fernando Bernal Cruz
Natalia Angélica Cancino García

Universidad Distrital Francisco José de Caldas
Facultad de Ingeniería, Ingeniería Electrónica
Bogotá D.C., Colombia
2016

RED DE SENSORES INALÁMBRICOS PARA MICRORREDES ELÉCTRICAS

Brayan Fernando Bernal Cruz
Natalia Angélica Cancino García

Trabajo de grado para optar al título de:

Ingeniero Electrónico

Director
Ing. Julián Rolando Camargo López

Co- Director
Ing. MSc. Elvis Eduardo Gaona

Grupo de Investigación

GITUD - Grupo de Investigación de Telecomunicaciones de la Universidad Distrital

Semillero de Investigación N-RED

Universidad Distrital Francisco José de Caldas
Facultad de Ingeniería, Ingeniería Electrónica
Bogotá D.C., Colombia
2016

Nota de aceptación

_____

_____

_____

_____

_____

_____

Director:
Ing. Julián Rolando Camargo López
Universidad Distrital Francisco José de Caldas

_____

Jurado: Luis Enrique Martin Santamaría
Universidad Distrital Francisco José de Caldas

Agradecimientos

Los autores del presente proyecto de grado queremos agradecer y reconocer a las siguientes personas e instituciones por su valiosa contribución:

A nuestro director Julián Rolando Camargo y codirector Elvis Eduardo Gaona por su apoyo constante y disposición para llevar a cabo este proyecto, de igual manera les damos gracias por brindarnos la oportunidad de pertenecer al grupo de investigación GITUD y al semillero de investigación N-RED.

A el Centro de Investigación y Desarrollo Científico de la Universidad Distrital por su apoyo incondicional para poder llevar a cabo este proyecto de investigación, innovación y creación.

A la Universidad Distrital Francisco José de Caldas, por haber permitido la formación académica para el desarrollo de nuestras capacidades durante nuestros años de estudio.

A nuestras familias por su compañía, por su incondicional apoyo, por sus consejos y por ayudarnos a ser mejores personas e impulsarnos en el día a día a ser excelentes profesionales.

A la Rama Estudiantil IEEE de la Universidad Distrital, a nuestros compañeros, a los profesores que nos formaron y a todos los que hicieron posible desarrollar este proyecto.

Resumen

El desarrollo de este proyecto de grado está enfocado en el diseño e implementación de un prototipo de una red de sensores inalámbricos para una mirorred eléctrica. Que sea capaz de medir las variables características de la red de distribución estableciendo un puente de comunicación con un nodo concentrador que permita la recepción de esta información adquirida.

Para el diseño de los nodos sensores se utilizaron elementos de medición de tensión y corriente de la marca LEM que trabajan con valores estándar de distribución de energía en Colombia. Para el procesamiento de las variables de tensión, corriente, frecuencia, fase y factor de potencia se utilizó la tarjeta ADSP de Texas instruments TMS320F28335. Para la enlace comunicación entre el nodo sensor y el nodo concentrador se utilizó dispositivos Xbee de la marca DIGI.

En el nodo concentrador se emplearon distintas herramientas informáticas como Netbeans y PostgreSQL, para la generación de un software que permitiera el almacenamiento y visualización de la información adquirida en los nodos sensores.

Se realizan pruebas individuales y luego de manera conjunta de cada uno de los módulos diseñados, para verificar el correcto funcionamiento y el cumplimiento de los objetivos propuestos.

Tabla de contenido

Agradecimientos 7
Resumen 9
Índice de graficas 15
Índice de tablas 19
Índice de ecuaciones 21
Índice de abreviaturas 23
GENERALIDADES 25
1.1. INTRODUCCIÓN 25
1.2. JUSTIFICACIÓN 26
1.3. OBJETIVOS 27
Objetivo General 27
Objetivos Específicos 27
GENERALIDADES DE LAS MICRORREDES ELÉCTRICAS 29
2.1. GENERACIÓN DISTRIBUIDA 29
2.2. MICRORRED 29
2.3. TOPOLOGÍA Y ELEMENTOS. 31
2.4. RED TRIFÁSICA. 33
2.5. ANÁLISIS DE POTENCIA PARA LA RED TRIFÁSICA. 36
2.6. ANÁLISIS DE INTERCONEXIÓN. 38
RED DE SENSORES INALÁMBRICOS 41
3.1 ELEMENTOS DE UNA RED DE SENSORES INALÁMBRICA. 41
3.2 ARQUITECTURA DEL NODO SENSOR 43
3.3 SISTEMA DE COMUNICACIÓN. 44
3.4 SISTEMAS DE MEDICIÓN INTELIGENTE O AVANZADA. 47
3.5 COMPONENTES DEL AMI 50
DISEÑO DE LA RED DE SENSORES INALÁMBRICO 55
4.1. CRITERIOS DE DISEÑO DEL NODO SENSOR 55
4.2. VARIABLES A MEDIR. 56
4.3. ELEMENTOS DE MEDICIÓN Y PRINCIPIO DE FUNCIONAMIENTO. 57

4.4. TOPOLOGÍA DE MEDICIÓN. 60
4.5. PROCESADOR DIGITAL DE SEÑAL (DSP). 60
4.7. HERRAMIENTAS DE SOFTWARE. 72
4.8. SISTEMA DE IDENTIFICACIÓN. 73
MODELAMIENTO Y SIMULACIÓN DE LA RED DE SENSORES INALÁMBRICOS 75
5.1. ACOPLE DE TENSIÓN. 75
5.2. ACOPLE DE FRECUENCIA. 77
5.3. ACOPLE DE CORRIENTE. 78
5.4. ACOPLE DEL ANGULO DE DESFASE. 79
5.5. ESQUEMA COMPLETO DE ACONDICIONAMIENTO DE SEÑALES. 81
5.6. PROCESAMIENTO Y CALCULO DE VARIABLES. 83
5.7. ENVIÓ Y RECEPCIÓN DE DATOS. 93
5.7. Diseño de circuitos impresos. 105
PRUEBAS Y RESULTADOS. 109
6.1. MODULO DE SENSADO. 109
6.2. SIMULACIONES 113
6.3. MODULO DE ACONDICIONAMIENTO. 116
6.4. MODULO DE PROCESAMIENTO. 119
6.5. MODULO DE COMUNICACIONES. 121
6.6. PROTOTIPO NODO SENSOR Y NODO CONTROLADOR. 129
CONCLUSIONES. 133
APORTES. 135
TRABAJOS FUTUROS. 137
REFERENCIAS. 139
ANEXOS. 145
11.1 DIAGRAMA ELÉCTRICO DE ESQUEMA DE SENSADO. 145
11.2 DIAGRAMA ELÉCTRICO DEL ACOPLE DE SEÑALES, TARJETA DE PROCESAMIENTO Y COMUNICACIÓN DEL NODO SENSOR. 146
11.3 DIAGRAMA ELÉCTRICO DE LA TARJETA DE ALIMENTACIÓN. 147
11.4 DIAGRAMA DE FLUJO DEL SOFTWARE GENERADO. 148
11.5 CÓDIGO PRUEBA PARA CALIBRACIÓN. 149
11.6 CÓDIGO PRUEBA CÁLCULO DE VALORES RMS. 152

11.7 CÓDIGO PRUEBA VALORES DE FRECUENCIA, FASE Y ÁNGULO DE DESFASE. 157
11.8 CÓDIGO PRUEBA ENVIÓ Y RECEPCIÓN DE DATOS. 168
11.9 CODIGO NODO 1. 173

Índice de graficas

Figura 1 Esquema general de una microrred. 30
Figura 2 Componentes generales de una microrred[21]. 32
Figura 3 Diagrama de conexión trifásica, izquierda Estrella derecha Delta. 33
Figura 4 Sistema trifásico de cuatro hilos con generador y carga conectado en estrella. 34
Figura 5 Grafica de Tensiones de Línea VL y Tensiones de Fase VF. 35
Figura 6 Grafica de Tensiones de Línea VL y Tensiones de Línea IL. 35
Figura 7 Grafica de Tensiones de fase VF y Corrientes de Línea IL. 36
Figura 8 Grafica en fasores de VL vs VF, VL vs IL (mencionadas de izquierda a derecha) 36
Figura 9 a) Triángulos de potencias, b) Factor de Potencia 38
Figura 10 Despliegue clásico de un WSN inteligente (adaptación de la revista IEEE Wireless Communications 2004) [14]. 42
Figura 11 Componentes de un nodo sensor (adaptación de la revista IEEE Wireless Communications 2004) [30]. 43
Figura 12 Tocologías de comunicación para una microrred acorde a la compatibilidad de los dispositivos [9]. 45
Figura 13 Jerarquía de una red inteligente (adaptación de la revista IEEE power & energy magazine 2010 [30]. 48
Figura 14 Jerarquía de una red inteligente (adaptación de la revista Electric Power Research Institute 2007) [45]. 50
Figura 15 Esquema de red hibrida (adaptación de revista Óptica Pura y Aplicada) [34]. 53
Figura 16 Esquema de red para labores de gestión(Adaptación) [49]. 54
Figura 17 Diagrama para el diseño de una red de sensores de una microrred 55
Figura 18 Principio de operación transductor de corriente, tomado del fabricante LEM. 58
Figura 19 Principio básico de funcionamiento, esquema de sensado y circuito equivalente (de izquierda a derecha). 58
Figura 20 Disposición de sensores en la red trifásica. 60
Figura 21 Base de acoplamiento de la familia Delfino C2000. 61
Figura 22 Tarjeta de Control de TMS320F28335. 61
Figura 23 Diagrama de bloques módulo ADC 63
Figura 24 Diagrama de bloques del sistema de múltiples módulos eCAP. 65
Figura 25 Diagrama de bloques módulo SCI. 67
Figura 26 Diagrama de bloques Timers. 68
Figura 27 Topologías de red 69
Figura 28 Capas del protocolo IEEE 802.15.4 70
Figura 29 Capas del protocolo IEEE 802.15.4 y Zigbee [52] 71
Figura 30 Acople sensor de tensión. 76
Figura 31 Esquema eléctrico del acople de tensión. 76
Figura 32 Nivel de referencia. 76

Figura 33 Acondicionamiento de señal de tensión para determinar la variable de frecuencia. 77
Figura 34 Esquema eléctrico del acople de frecuencia. 77
Figura 35 Acople sensor de corriente. 78
Figura 36 Esquema eléctrico del acople de corriente 78
Figura 37 Acondicionamiento de señal de corriente para genera PWM. 79
Figura 38 Esquema eléctrico del acondicionamiento de la señal de corriente para generar el PWM. 80
Figura 39 Obtención y acondicionamiento de señal PWM del desfase entre la tensión y corriente. 81
Figura 40 Señal de desfase entre la tensión y la corriente. 81
Figura 41 Esquema general de acondicionamiento de señales para el nodo sensor. 82
Figura 42 Diagrama de flujo prueba para calibración 85
Figura 43 Diagrama de flujo prueba cálculo de valores RMS 88
Figura 44 Captura por detección flanco de subida y modo Absolute time-stamp. 89
Figura 45 Captura por detección flanco de subida / bajada y modo Absolute time-stamp. 89
Figura 46 Diagrama de bloques prueba valores de frecuencia, fase y ángulo de desfase. ... 92
Figura 47 Esquema de conexión entre el dispositivo de comunicación y la tarjeta de procesamiento. 93
Figura 48 Diagrama de flujo prueba envío y recepción de datos 94
Figura 49 Trama a enviar 95
Figura 50 Esquema general de procesamiento y comunicación del nodo sensor 97
Figura 51 Esquema de conexión de tarjeta de procesamiento. 98
Figura 52 Base de datos generada en el PostgreSQL 100
Figura 53 Interfaz de Puerto de conexión con la red 101
Figura 54 Interfaz de consulta 102
Figura 55 Interfaz de petición. 102
Figura 56 Interfaz de visualización. 103
Figura 57 Interfaz completa de la aplicación. 103
Figura 58 Diagrama de flujo del software del nodo concentrador. 104
Figura 59 Diseño del circuito impreso de la tarjeta de sensado. 105
Figura 60 Tarjeta de sensado vista 3D. 106
Figura 61 Diseño del circuito impreso de la tarjeta de adquisición de información y comunicación. 107
Figura 62 Tarjeta de adquisición de información y comunicación vista 3D. 107
Figura 63 Diseño del circuito impreso de la tarjeta de alimentación. 108
Figura 64 Tarjeta de alimentación vista 3D. 108
Figura 65 Fuente trifásica regulada LEYBOLD 725702 109
Figura 66 Reóstatos de 10 ohm PHYWE 110
Figura 67 Señales de sensado, para una tensión de 85V y una corriente de 4A. 110
Figura 68 Comportamiento lineal del Sensor de tensión. 111
Figura 69 Comportamiento lineal del Sensor de corriente. 112
Figura 70 Montaje de los circuitos de sensado y acople. 113

Figura 71 Simulación del acople de tensión, azul señal simulada del sensor de tensión, rojo señal de salida acoplada. 114
Figura 72 Simulación del acople de frecuencia, azul señal simulada del sensor de tensión, rojo señal de PWM generada. 114
Figura 73 Simulación acople de corriente, azul señal de entrada, rojo señal acoplado. 115
Figura 74 Simulación del circuito de acople del desplazamiento de la señal de tensión y corriente. 115
Figura 75 a. Señal Acoplada Tensión b. Señal Acoplada Corriente 116
Figura 76 Señal de frecuencia, rojo frecuencia señal de tensión, azul PWM de corriente desplazada. 117
Figura 77 Módulo de cargas inductancias IGT serie 078 118
Figura 78 Señal de PWM correspondiente a la distancia de desplazamiento. 118
Figura 79 Tramas recibidas y almacenadas 124
Figura 80 Prueba realizada a 13 m 125
Figura 81 Grafica range test 126
Figura 82 Range test 126
Figura 83 Grafica prueba de rendimiento 127
Figura 84 Prueba de rendimiento 127
Figura 85 Montaje del prototipo. 129
Figura 86 Consola de Netbeans 130
Figura 87 Panel completo del Software desarrollado. 131
Figura 88 Graficas generadas por el software a partir de la información sensada de la microrred. 131

Índice de tablas

Tabla 1 Tecnologías y protocolos de comunicación [36] [31],[37],[38] 46
Tabla 2 Sensor LV-25 P LEM. 59
Tabla 3 Sensor LTSR-25 LEM. 60
Tabla 4 comparativa XBee Serie 1 vs Serie 2 70
Tabla 5 Campo Tipo de valor y su correspondiente número identificador 95
Tabla 6 Identificación de valores calculados 100
Tabla 7 Tabla de variables y tipo, para la base de datos 100
Tabla 8 Salida Sensor de tensión. 111
Tabla 9 Salida Sensor de Corriente. 111
Tabla 10 Salida Acople de tensión. 117
Tabla 11 Salida Acople de corriente 117
Tabla 12 Resultados prueba calibracacion con una tensión de entrada de 100 V 119
Tabla 13 Resultados prueba calibración con una tensión de entrada de 50v 119
Tabla 14 Resultados prueba calibración con una corriente de entrada de 4 A 120
Tabla 15 Resultados prueba calibración con una corriente de entrada de 2 A 120
Tabla 16 Resultados prueba de tensiones RMS 120
Tabla 17 Resultados prueba de corrientes RMS 121
Tabla 18 Resultados prueba de frecuencias 121
Tabla 19 Resultados prueba de fases 121
Tabla 20 Resultados prueba de ángulos de desfases entre VL vs IL 121
Tabla 21 Configuración módulos XBEE Serie 2 topología estrella 122
Tabla 22 Tramas enviadas al nodo concentrador 123
Tabla 23 prueba de transmisión y recepción del sistema con diferentes distancias 125
Tabla 24 Resultados range test de los módulos Xbee 127
Tabla 25 Resultados prueba de rendimiento de los módulos Xbee 128

Índice de ecuaciones

Ecuación 1 Ecuación de tensión y corriente de línea. 35
Ecuación 2 Ecuaciones de potencia de la red trifásica [23]. 37
Ecuación 3 Conversión del módulo ADC. 64
Ecuación 4 Relación del registro del reloj 83
Ecuación 5 Configuración del periodo. 84
Ecuación 6 pendiente entre dos tensiones traducidos 86
Ecuación 7 tensión de conversión 86
Ecuación 8 pendiente entre dos corrientes traducida 86
Ecuación 9 corriente de conversión 86
Ecuación 10 Tensión DC. 87
Ecuación 11 Tensión RMS con nivel DC 87
Ecuación 12 Tensión RMS 87
Ecuación 13 Corriente DC 87
Ecuación 14 Corriente RMS con DC. 87
Ecuación 15 Corriente RMS 87
Ecuación 16 Calculo de Frecuencia. 90
Ecuación 17 Calculo de Fase. 91
Ecuación 18 Calculo de ángulo de desfase 91
Ecuación 19 Ecuaciones de potencia por línea. 99

Índice de abreviaturas

AC	CORRIENTE ALTERNA
DC	CORRIENTE DIRECTA
CL	LAZO CERRADO
VF	TENSIÓN DE FASE
IL	CORRIENTE DE LÍNEA
VL	TENSIÓN DE LÍNEA
VLL	TENSIÓN DE LÍNEA
IL	CORRIENTE DE LÍNEA
WSN	RED DE SENSORES INALÁMBRICOS
AMI	SISTEMA DE MEDICIÓN AVANZADA
AMR	LECTURA DE MEDICIÓN AUTOMÁTICA
PWM	MODULACIÓN POR ANCHO DE PULSO
ADC	CONVERSIÓN ANÁLOGA DIGITAL
MDM	SISTEMA DE GESTIÓN DE DATOS MEDIDOS
SAP	SERVICE ACCESS POINT
ADC	CONVERSOR ANÁLOGO DIGITAL
MVCC	MULTI-VERSIÓN DE CONCURRENCIA
DSPs	PROCESADOR DIGITAL DE SEÑALES
ASIC	CIRCUITOS INTEGRADOS DE APLICACIÓN ESPECÍFICA
FPGA	DISPOSITIVO LÓGICO PROGRAMABLE

1
GENERALIDADES

1.1. INTRODUCCIÓN

El desarrollo tecnológico ha permitido que hoy en día las comunicaciones sean más rápidas y eficientes, esto unido a la evolución en tecnologías inalámbricas, han permitido ofrecer mayor cobertura y capacidad para transferir datos al momento de establecer una mejor conexión, haciendo más flexible el despliegue de las redes logrando su escalabilidad.

El flujo de información que se transmite por estos medios inalámbricos hoy en día, es enorme y se hace imperativo que existan sistemas capaces de satisfacer estas necesidades, surgiendo así sistemas que transmiten una gran cantidad de información, con una tasa de pérdida de muy baja. El alcance de estas tecnologías, se ha desarrollado dependiendo de la necesidad que se quiere cubrir dando paso a nuevas aplicaciones, algunos sistemas de transmisión como el infrarrojo, WPAN, BlueTooth, ZigBee, Wi-Fi, WLAN etc. fueron expandiéndose en diferentes áreas según sus características, y se diferencian por su limitación de alcance y velocidad de transmisión.

Debido a la gran variedad de aplicaciones que se pueden llegar a tener con las redes inalámbricas, se inició la incursión en diferentes campos de trabajo como las ciudades inteligentes, de allí surgió el desarrollo de redes de sensores inalámbricos (WSN, Wireless Sensor Networks). Las aplicaciones que pueden llegar a tener estas redes de sensores, abarcan el sector agrícola, ambiental, la domótica, la seguridad, las comunicaciones, en la medicina, logística, todo tipo de controles y monitoreo, aplicaciones militares y ahorro energético [1], [2], [3], [4]. Esta tecnología es muy apetecida ya que hace el uso de sensores de bajo costo, se puede obtener información del entorno, procesarla y transmitirla por medios inalámbricos[5], [6].

La obtención, procesamiento y transmisión de la información adquirida converge en una infraestructura de red de sensores inalámbrico aplicado a microrredes, que se convierte en el elemento fundamental para la creación de una red inteligente de sensores que permite realizar el monitoreo de una microrred, teniendo como base la comunicación inalámbrica entre dispositivos para finalmente observar el comportamiento de forma remota y fiable [7], [8]. También proporciona a las compañías de energía eléctrica, una mejora en la comprobación del pago del servicio para efectuar un corte de luz. Una Infraestructura de Medición Avanzada (AMI, Advance Metering Infrastructure) es el único sistema donde son detectables todos los segmentos de línea y subestaciones en un sistema completo. Esta no sólo es utilizada para el control de corte de luz, sino que también permite a las compañías eléctricas, comprobar los clientes que requieren la restauración del servicio [9], permite a los consumidores a tomar decisiones en tiempo real sobre la utilización de energía. El uso de AMI permite a las empresas de servicios públicos aumentar la eficiencia de las redes de energía regionales mediante la gestión de carga de la demanda durante las horas pico y la

reducción de la generación de energía innecesaria [10]. Permitiendo optimizar los sistemas de generación tradicionales y modernos basados en energías renovables.

Las condiciones sociales y ambientales han llevado a la necesidad de un desarrollo sostenible, afectando de manera directa la forma de generación de energía eléctrica, dando paso a nuevos sistemas de obtención como lo son los sistemas fotovoltaicos, plantas eólicas, entre otros. Lo que ha provocado cambios en las microrredes por lo que hace necesario el control de los nuevos sistemas generadores a una red de distribución de energía ya estructurada [11].

La integración de los sistemas de sensores inalámbricos y los sistemas de microrredes permiten monitorear todos estos cambios en la red, desde el punto de generación al punto de consumo, para poder realizarlo es necesario realizar la medición de diferentes variables y el análisis de las mismas [12].

Diversas circunstancias pueden cambiar los parámetros de la red, como las fluctuaciones de la demanda, defectos del transformador, la aparición de armónicos, perturbaciones atmosféricas, etc. Incluso la interrupción del suministro de energía en un periodo de tiempo corto, menor a un segundo, es suficiente para llevar a las máquinas modernas a un punto muerto, lo que resulta en horas de producción interrumpida y desechos irrecuperables de material de producción.

La necesidad de obtener todos los parámetros posibles en la red trifásica o monofásica, ya sea de una instalación industrial (trifásico) o suministro de corriente doméstica (monofásico) ha dado lugar al desarrollo de diferentes sistemas de captura de datos. En un comienzo los dispositivos se basaban en principios electromecánicos; Actualmente existen sistemas electrónicos disponibles para llevar a cabo las funciones necesarias sobre la base de una conversión analógico-digital (ADC) en el que los cálculos se realizan digitalmente con mayor precisión [13].

1.2. JUSTIFICACIÓN

Las microrredes eléctricas se han convertido en una alternativa para la disminución del uso de los sistemas de generación eléctrica tradicionales, sobretodo porque permite la inclusión de nuevos métodos de generación que son amigables con el medio ambiente. Poder conocer el comportamiento de estos sistemas es de suma importancia para optimizar el uso de los recursos disponibles.

La necesidad de obtener todos los parámetros posibles de una red eléctrica ha dado el desarrollo de diferentes sistemas de captura de datos que carecen de un análisis de las necesidades en sitio y sus técnicas de adquisición muchas veces son rudimentarias, mostrando retardos, perdidas de los datos al momento de realizar la adquisición en el nodo sensor. Se presenta la red de sensores inalámbricos como una alternativa de adquisición, concentración y visualización para identificar el comportamiento de la red eléctrica. Este proyecto está enfocado hacia el desarrollo de un prototipo de una red de sensores inalámbricos para una microrred, adquiriendo las distintas variables características de la red trifásica en nodos sensores y que estas puedan ser consultadas en el nodo concentrador.

1.3. OBJETIVOS

Objetivo General

Implementar una red de sensores inalámbricos WSN (Wireless sensor Network) que realice la medición de diversas variables eléctricas de una microrred, las centralice y visualice en un nodo central.

Objetivos Específicos

- Realizar el análisis de requerimientos en términos de comunicaciones para el diseño e implementación de la red de sensores inalámbricos que opere en una microrred
- Diseñar el dispositivo electrónico que realice la medición, procesamiento y transmisión de los datos de voltaje, corriente, frecuencia y fase de una microrred eléctrica.
- Implementar la red de sensores inalámbricos que transmita la información de los parámetros medidos en la microrred, atendiendo los niveles de calidad de servicio de la red inalámbrica adecuados.
- Desarrollar la herramienta computacional que permita visualizar el comportamiento de la generación de energía, el consumo en la microrred y/o las variables medidas.

2
GENERALIDADES DE LAS MICRORREDES ELÉCTRICAS

2.1. GENERACIÓN DISTRIBUIDA

El concepto de Generación Distribuida es relativamente nuevo, ha surgido de la evolución que ha tenido el sistema eléctrico, a pesar de ser un concepto en desarrollo se puede definir como la generación de energía eléctrica a pequeña escala y a cortas distancias del usuario [14] [15].

Está conformada por varios sistemas generadores que alimentan diversas cargas determinadas y en dado caso de que uno de los sistemas generadores no pueda entregar la energía eléctrica necesaria, entran a operar otros generadores conectados en la red para suplir la energía, suministrando de esta forma la energía requerida a la carga de manera ininterrumpida.

2.2.MICRORRED

Las microrredes, conocidas de igual manera como "Microgrids", es una red de distribución de energía relativamente pequeña, basan su funcionamiento en que la generación y el consumo de energía no tienen que ser sistemas independientes, sino todo lo contrario, pueden hacer parte del mismo sistema, con diferente estructura al sistema tradicional, en el cual el punto de generación eléctrica se encuentra totalmente apartado del consumo o de los usuarios, funcionando de esta manera como dos sistemas independientes [16].

Una microrred está ubicada en el área de media tensión (MV - Médium Voltage) y baja tensión (LV - Low Voltage) [16]. Está compuesta por un conjunto de recursos distribuidos de energía, tales como, generadores, cargas, sistemas de control y dispositivos de almacenamiento de energía [10]. Una microrred normalmente está conectada al sistema eléctrico de la generadora de energía, pero tiene la capacidad de operar de forma aislada en un eventual caso [17]. Por ende, tiene la capacidad de entregar y recibir energía de la red sin ningún problema, controlando el flujo de energía [18].

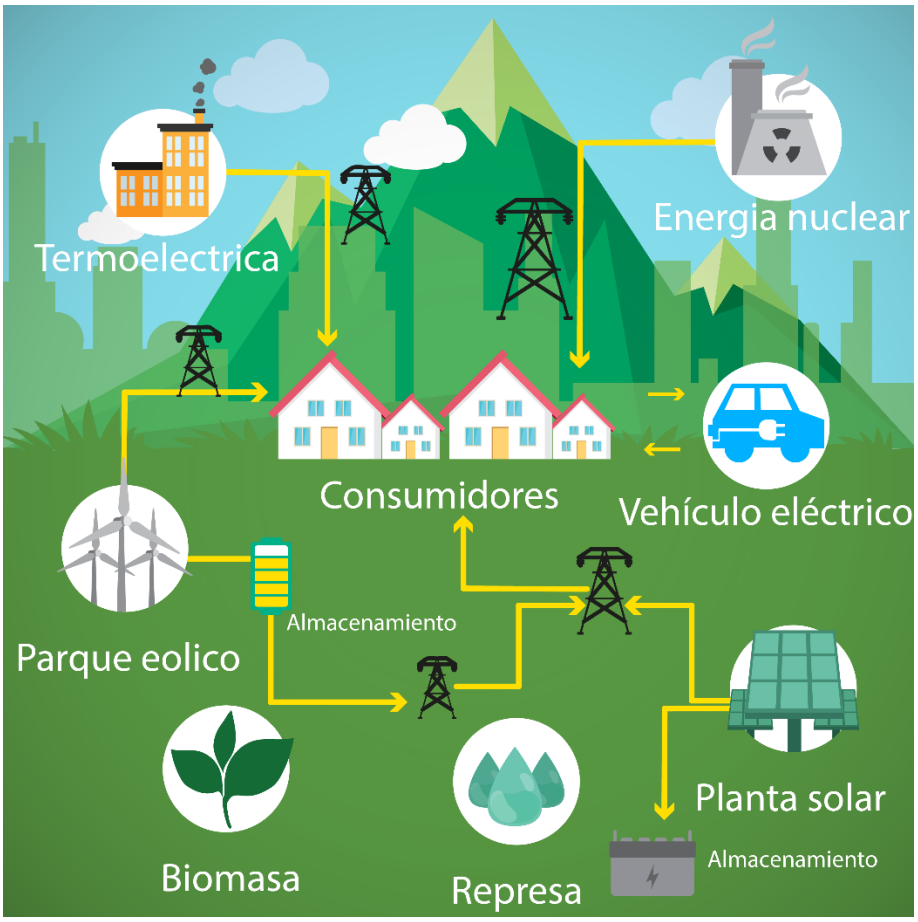


Figura 1 Esquema general de una microrred.

Las microrredes deben proporcionar energía con la fiabilidad con la que funciona en los casos de interrupción de energía por parte del ente encargado de entregar la energía tradicional. En la figura 1 se puede observar un breve esquema de una microrred.

Los beneficios de este sistema son diversos los más destacados son económicos y ambientales. Gran parte de las investigaciones que se realizan en todo el mundo están enfocadas en el desarrollo sostenible, es aquí donde el uso de fuentes de energía renovable se hace fundamental, las microrredes permiten la integración de fuentes de generación de energía que tienen una baja emisión de carbono, como la eólica, la energía solar o fotovoltaica, biomasa renovable, hidrógeno entre otras; la inclusión de estos nuevos formas de generación afectara el mercado y la organización del sistema energético [19].

Adicionalmente permiten evitar costos en la infraestructura de transmisión. Lo cual presenta también, una solución práctica para la generación de energía en zonas remotas que actualmente no tienen servicio de electricidad o el mismo es discontinuo [20].

2.3.TOPOLOGÍA Y ELEMENTOS.

Las microrredes no tienen una topología específica, pero si se puede identificar algunas características que resaltan en su estructura y se vuelven importantes para determinar su comportamiento.

Se pueden encuentran normalmente en dos modos de trabajo, el primero es cuando están conectadas al sistema de distribución eléctrica tradicional y el segundo cuando operan de manera aislada o sin conexión alguna con la red de distribución. Esto resulta ser una ventaja de las microrredes, no se limitan a trabajar únicamente conectadas al sistema de distribución sino también pueden trabajar de manera aislada sin ningún inconveniente.

- Microrred conectada al sistema de distribución.

En condiciones normales de funcionamiento, una microrred estará conectada a la red de distribución eléctrica. En esta situación no es necesario que toda la energía demandada por las cargas sea producida por los generadores de la microrred ya que el desfase entre generación y consumo será cubierto por la energía que fluye a través del punto de conexión.

La microrred podrá ser vista como un pequeño generador o como una carga en función de si la generación es mayor o menor que la energía demandada por las cargas. Cuando la microrred trabaja conectada a la red de distribución, el sistema de control tiene como objetivo la disminución de coste de la energía para los consumidores que están asociados a ella. Usa la generación local cuando es económicamente rentable, disminuyendo la cantidad de energía que toma de la red.

- Microrred en modo aislado.

Una microrred estará habitualmente (modo normal de operación) conectada a la red de distribución. Sin embargo, ante un fallo en esta red, es posible pasar al modo de emergencia en que la microrred se desconectase de la red y pase a trabajar de forma autónoma. Este modo de trabajo es permanente cuando la microrred se encuentra en regiones apartadas en donde el acceso a la red de distribución es imposible.

Una vez detectado un problema en la red de distribución, se tratará siempre de que la desconexión se realice de la forma más transparente posible para las cargas conectadas. Si no es posible aislar la microrred con éxito y el suministro queda afectado, se pone en marcha un procedimiento automático con el objetivo de restaurar el servicio de forma independiente de la red de distribución.

Esta característica de modo de conexión de la microrred está orientada a la demanda de generación de energía por el método tradicional o alternativos, en donde se busca es asegurar la operación de la red de manera continua sin importar la usencia de alguno o de varias formas de generación. En la figura 2 se muestran un esquema general de una microrred con algunos de sus componentes más comunes.

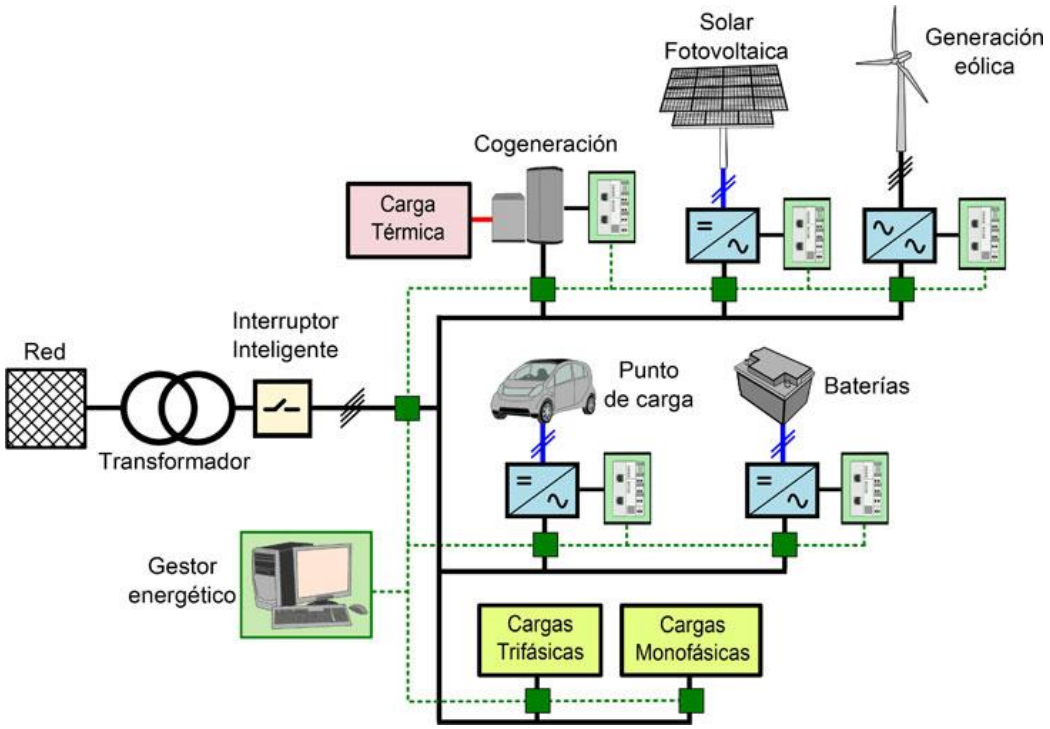


Figura 2 Componentes generales de una microrred[21].

Se pueden identificar los elementos que la conforman en los siguientes ítems:

- Generadores, en esta categoría entran sistemas alternos, tradicionales o la red en general.
- Cargas, estas pueden ser de tipo trifásicas, monofásicas o incluso continúas dependiendo de la necesidad que se quiera suplir.
- Almacenamiento, son bancos de baterías que se encargar de almacenar la energía para hacer frente a posibles desconexiones del sistema.
- Convertidores, estos pueden ser de tipo corriente continua (CC)-corriente alterna CA, CC-CC o CA-CA, tienen la función de convertir el tipo de señal para poder interconectarla a otro punto de la red.
- Conexión o bus, se habla de líneas de conexión continua y alterna.

Al realizar el análisis del modo de conexión de la microrred e identificando sus cualidades, se puede determinar que sin importar el modo de operación del sistema la característica fundamental es la conexión de los distintos métodos de generación y las cargas. Si se desea entregar energía a la red de distribución esta debe compartir las características de la red, por estándar la distribución de energía a nivel mundial que se realiza comercialmente es de tipo alterna, variando niveles de tensión y frecuencia.

En la figura 2 se puede observar un esquema donde se evidencia las diferentes líneas de conexión, CC y CA, como en el punto de carga para un automóvil y en la generación fotovoltaica, sin importar el método de generación alternativo que se prefiera la energía que

se entrega al usuario es alterna, por lo cual si se genera de tipo continuo es necesario emplear dispositivos inversores que conviertan esta energía.

Siendo finalmente esta la característica más relevante al momento de conocer el comportamiento de la microrred, la distribución de corriente alterna puede darse de modo monofásico, bifásico o trifásico. Por compatibilidad de dispositivos ya que es el más frecuente, económico y estándar en la distribución, se opta por el diseño de un sistema de sensado para redes trifásicas [22].

2.4.RED TRIFÁSICA.

Los sistemas trifásicos son importantes por distintas razones, primero, casi toda la potencia eléctrica que se genera y distribuye en todo el planeta es en forma trifásica, a una frecuencia de 60 Hz o de 50 Hz dependiendo la parte del mundo. Cuando se requiere el uso de otras entradas alternas como las monofásicas o polifásicas en general, se toman del sistema trifásico y se adecuan a la necesidad. Segundo, la potencia instantánea es constante produciendo una transmisión de potencia uniforme. Tercero la cantidad de alambre requerida para un sistema trifásico es menor que la que requiere in sistema monofásico del mismo equivalente [22].

El análisis de una red trifásica se puede realizar desde un generador o alternador, el cual tiene un devanado estacionario o un estator en el que se disponen físicamente bobinas independientes a 120° de distancia entre sí y un rotor. Cuando esta gira se genera un campo magnético que induce tensión en las bobinas, la tensión generada es igual en las tres bobinas y se encuentran desfasadas entre si 120°. La fuente de tensión generada se puede conectar en modo delta y en modo estrella.

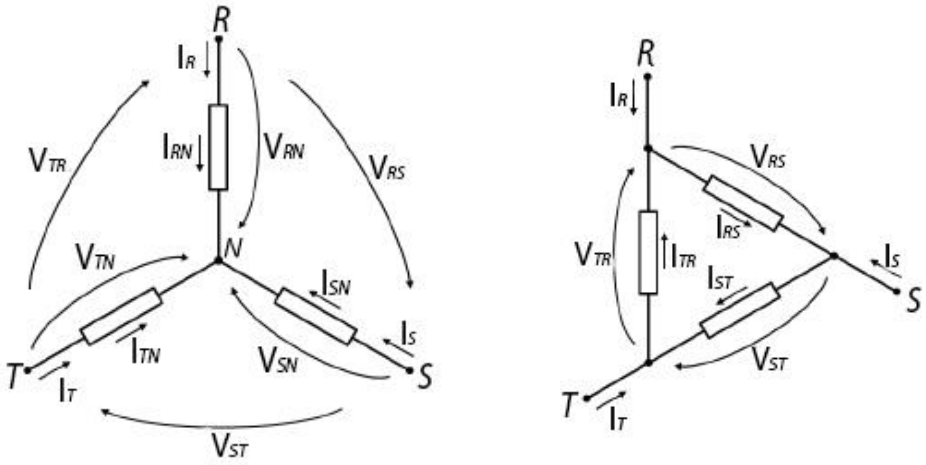


Figura 3 Diagrama de conexión trifásica, izquierda Estrella derecha Delta.

Existen distintas combinaciones de conexión entre generación y carga utilizando estos dos diagramas, donde el primer símbolo hace referencia a la conexión de generación y el segundo a la conexión de carga:

- Conexión Estrella-Estrella (Y-Y).
- Conexión Delta-Detal (Δ- Δ)
- Conexión Estrella-Delta (Y- Δ)
- Conexión Delta-Estrella (Δ-Y)

El análisis del sistema de conexión Y-Y, la figura 3 muestra un sistema trifásico de cuatro hilos con generador y carga conectado en estrella, es clave para los sistemas trifásicos puesto que cualquier sistema puede reducirse a un sistema de este tipo. El análisis partirá con un sistema balanceado, es decir su fuente y carga están balanceadas. Una carga balanceada es aquella en que las impedancias de las fases son iguales en magnitud y fase.

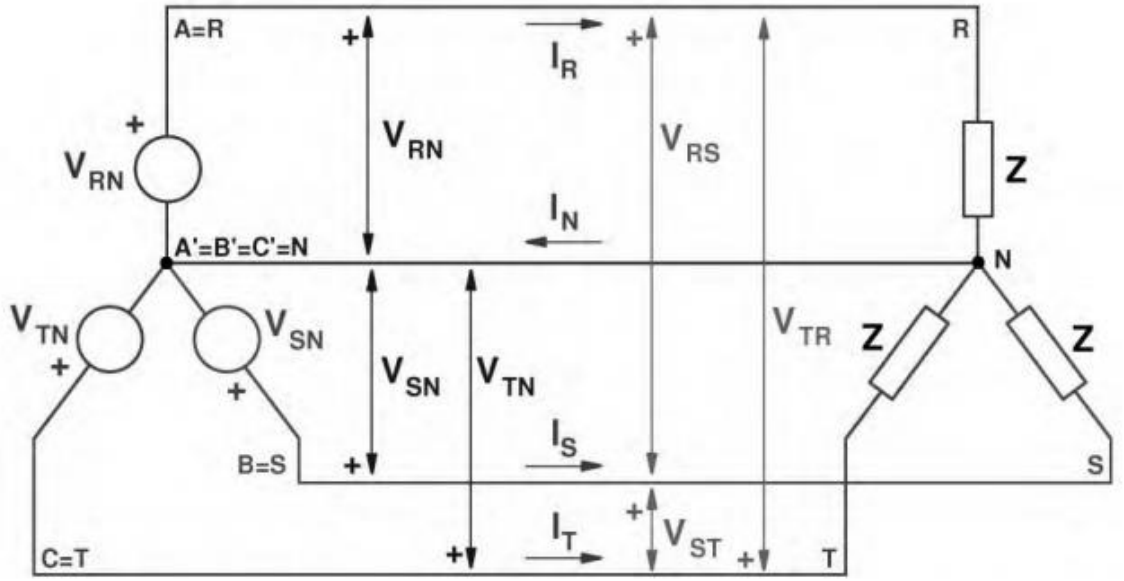


Figura 4 Sistema trifásico de cuatro hilos con generador y carga conectado en estrella.

En la figura 4 se presenta un sistema de cuatro hilos donde los conductores de las fases son respectivamente R, S y T que son los encargados de llevar la energía generada a la carga. El conector neutro identificado con la letra N, no tendrá gran participación en el análisis del sistema balanceado incluso es eliminado en diagramas de distribución de energía y se hace un sistema de aterrizaje diferente puesto que se asegura que el sistema sea completamente balanceado, pero tiene una participación muy significativa en los sistemas desbalanceados.

Las tensiones *VFR-N VFS-N* y *VFT-N* se encuentran respectivamente entre las líneas R, S, T y la línea de neutro (n), son denominas tensiones de fase. Las tensiones *VLR-S, VLS-T* y *VLT-R* son llamadas de línea-línea (*VL*) y están relacionadas con las tensiones de fase (*VF*). Las corrientes de línea (*VL*) en los sistemas Y-Y resultan ser las mismas corrientes de fase del sistema (*VF*). Esto se puede ver más claro mediante la ecuación 1.

$$VL = VP * \sqrt{3} \quad \theta = 30^0 \qquad IL = IF$$

Ecuación 1 Ecuación de tensión y corriente de línea.

En el caso de las tensiones el ángulo de 30° que aparece se debe a que, las tensiones de línea se adelantan respecto a las tensiones de fase, esto se puede observar en la figura 5. Esto implica que la corriente de línea también se encontrara con una distancia de 30° de la tensión de línea, se aprecia en la figura 6. La corriente de línea y la tensión de fase se encuentran en sincronía como se observa en la figura 7, esto se puede observar de igual manera mediante el diagrama de fasores, como se presenta en la figura 8.

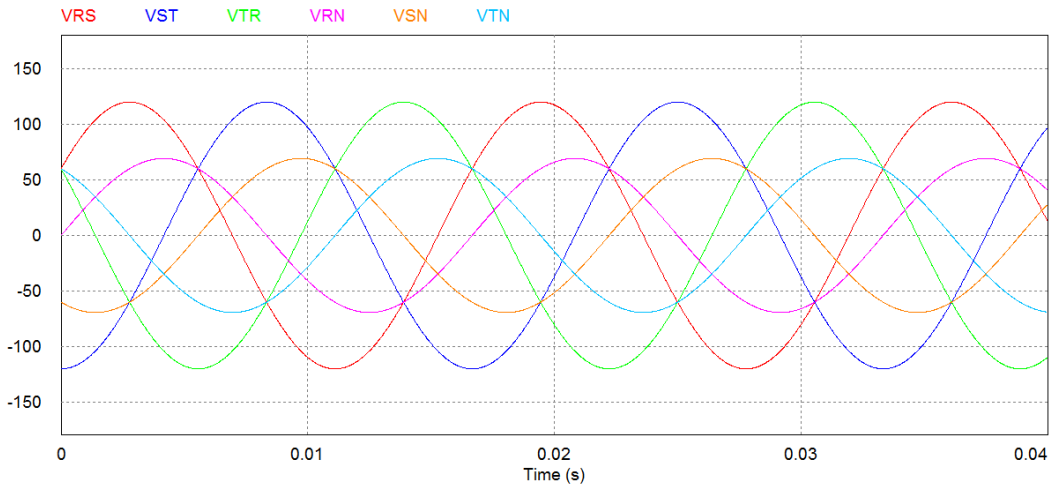


Figura 5 Grafica de Tensiones de Línea VL y Tensiones de Fase VF.

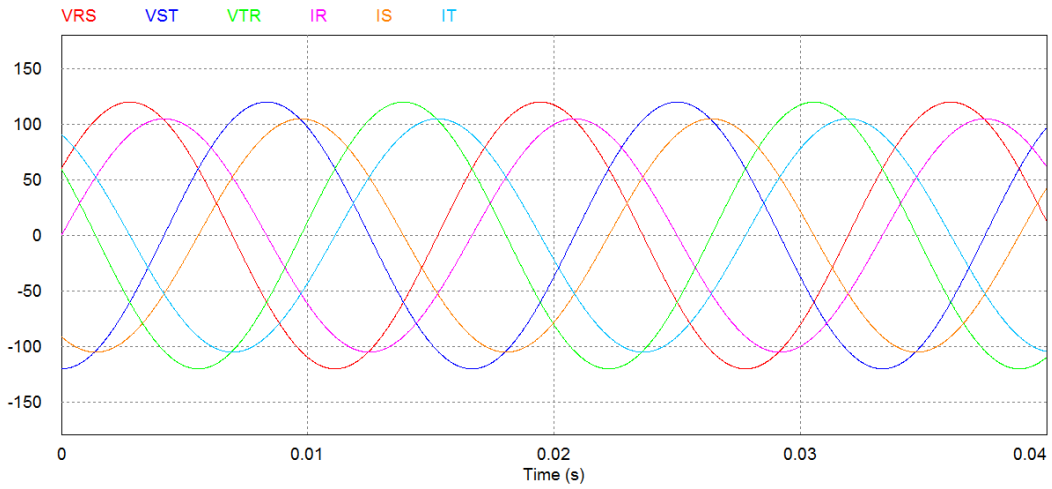


Figura 6 Grafica de Tensiones de Línea VL y Tensiones de Línea IL.

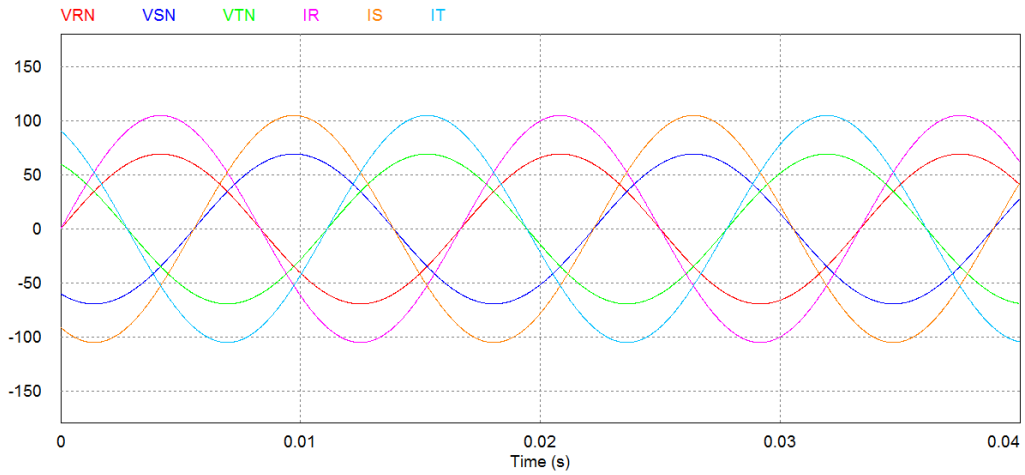


Figura 7 Grafica de Tensiones de fase VF y Corrientes de Línea IL.

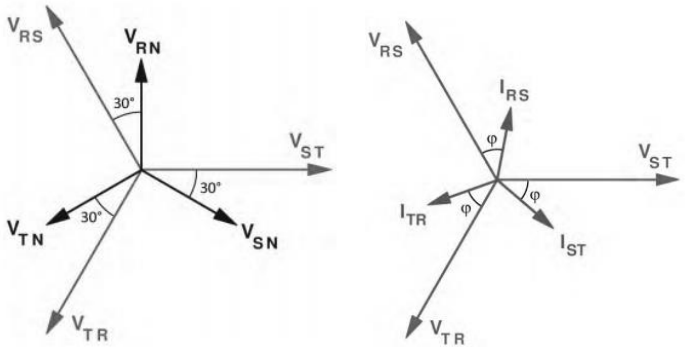


Figura 8 Grafica en fasores de VL vs VF, VL vs IL (mencionadas de izquierda a derecha)

2.5. ANÁLISIS DE POTENCIA PARA LA RED TRIFÁSICA.

Una carga balanceada es aquella en que las impedancias de las fases son iguales en magnitud y fase. Cuando esto ocurre las corrientes y tensiones de cada una de las cargas son iguales, cuando la carga no es un sistema equilibrado se generan distintos valores de corriente y según la naturaleza de la carga se generan desplazamientos en la señal, la distancia desplazada o el ángulo desplazado de la señal de corriente respecto la señal de tensión, es el factor adicional que permite calcular de manera correcta la potencia consumida por el sistema [22].

En los circuitos de corriente alterna se hace necesario hablar no solo de potencia instantánea, sino también de potencia activa, reactiva y aparente [23], [22].

- Potencia activa (P), es la que se aprovecha como potencia útil. También es conocida como potencia media, real o verdadera y es causada por los elementos resistivos. Su unidad de medida en el vatio (W).
- Potencia reactiva (Q), que es la potencia que usa los elementos inductivos y los capacitivos para generar campos magnéticos o eléctricos, esta potencia no se transforma en trabajo efectivo, sino que oscila por la red. La potencia reactiva es positiva si el receptor es inductivo y negativa si el receptor es capacitivo, coincidiendo en signo con la parte imaginaria de la impedancia. Su unidad de medida es el voltamperio reactivo (V Ar).
- Potencia aparente (S), esta es la potencia total consumida y es el producto de los valores eficaces de tensión e intensidad. Se obtiene como la suma vectorial de las potencias activa y reactiva y representa la ocupación total de la instalación debida a la conexión del receptor. Su unidad de medida es el voltamperio (V A).

Existen diferentes métodos para obtener las ecuaciones que relacionan las variables, de manera concreta las ecuaciones resultantes para el cálculo de la potencia se presentan en la ecuación 2.

$$P = \sqrt{3} V\mathrm{L} * I\mathrm{L} * \cos \emptyset$$

$$Q = \sqrt{3} V\mathrm{L} * I\mathrm{L} * \mathrm{sen}\, \emptyset$$

$$S = \sqrt{3} V\mathrm{L} * I\mathrm{L}$$

Ecuación 2 Ecuaciones de potencia de la red trifásica [23].

Estas ecuaciones de potencia se aplican para los sistemas trifásicos balanceados, para los circuitos desbalanceados se elimina el factor de multiplicación $\sqrt{3}$ y se realiza el análisis en cada una de las líneas.

Una representación común es el triángulo de potencia, el cual se ilustra en la figura 9a, el triángulo contiene cuatro elementos, la potencia aparente, la potencia activa, la potencia reactiva y el ángulo del factor de potencia, como se muestra en la figura 9b.

La potencia aparente (S) contiene toda la información de potencia de una carga, la parte real es la potencia activa (P), su parte imaginario representa la potencia reactiva (Q) y el coseno de su ángulo es el factor de potencia (fp). Cuando la potencia aparente se ubica en el primer cuadrante se tiene una carga de tipo inductivo y un factor de potencia atrasado, cuando está en el cuarto cuadrante, la carga es capacitiva y el factor de potencia está en adelanto.

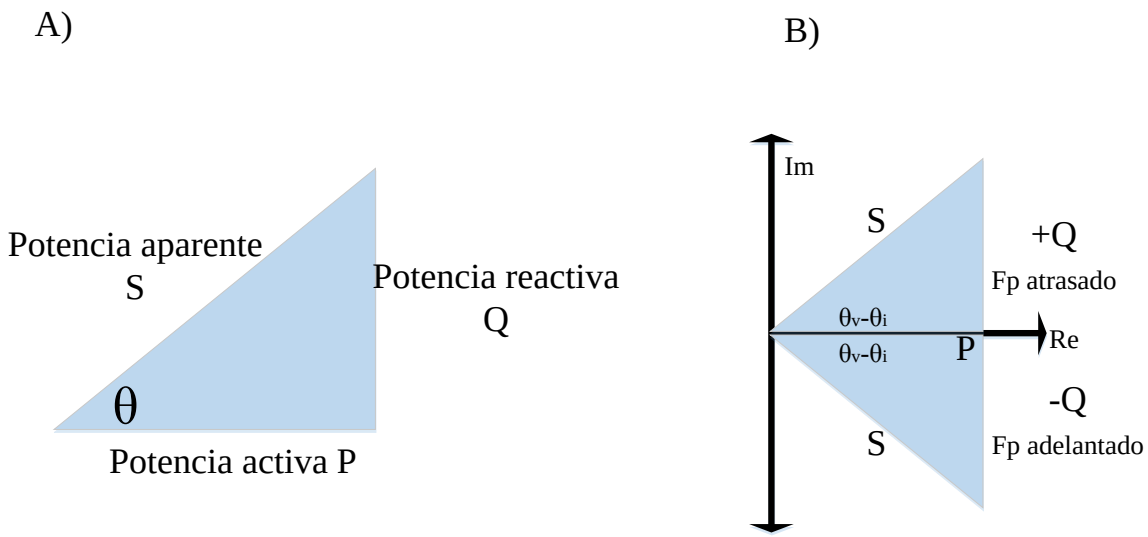


Figura 9 a) Triángulos de potencias, b) Factor de Potencia

2.6.ANÁLISIS DE INTERCONEXIÓN.

Como se mencionaba anteriormente, una de las características de las microrredes es la integración de diferentes métodos de generación de energía eléctrica, al realizar el proceso de interconexión de los dispositivos se debe prestar especial atención a las características de la señal generada por cada generador.

Es necesario asegurar el acople de las señales entre sí para poder realizar una correcta conexión sin averiar ningún elemento de la red, en la sección anterior se menciona un análisis eléctrico de la microrred partiendo de que su característica más representante es el flujo de energía mediante líneas de corriente alterna, haciendo un breve énfasis en las magnitudes de tensión y corriente del sistema.

Cuando se tiene un solo generador y una carga, el flujo de energía se puede decir que va de la generación a la carga sin ningún componente extra, al incluir más generadores que alimentan la misma carga aparece un punto intermedio en este flujo de energía y es el punto de interconexión de elementos.

Cuando se habla de corriente continua este punto de conexión representa la suma algebraica de las distintas variables modificando así valores de tensión y corriente, cuando se tiene sistemas de corriente alterna también se representa como la red continua, pero aparecen un concepto importante que deben analizarse que es la sincronización de los dispositivos

- Sincronización.

La sincronización de la red consiste en que la señal que va ser entregada por el generador coincida con la carga mediante las características que se encuentren en los estándares que adopten en el lugar donde se encuentre la microrred.

Esto conlleva a que aparezca la frecuencia como variable de gran interés, si se llegara a presentar una variación o algunas perturbaciones entre las señales generadas puede repercutir en distintos problemas para el sistema, afectando de manera directa su calidad [24].

Independientemente de la fuente de generación, los inversores están conectados a la red eléctrica se caracterizan en su mayoría por operar como fuentes de corriente en fase a la misma frecuencia de la red [25]. Estos dispositivos son los directamente encargados de controlar la sincronización, de igual forma es conveniente realizar un seguimiento a estas variables.

3
RED DE SENSORES INALÁMBRICOS

En 1961 la marina de Estados Unidos instalo una red de sensores inalámbricos en las profundidades de los océanos Atlántico y Pacifico, con el fin de tener un sistema de vigilancia para la detención de amenazas marítimas. El sistema funcionaba a través de la producción de señales acústicas, que detectaban submarinos enemigos [26].

Las nuevas tecnologías han permitido que en la fabricación de los sensores disminuyera sus costos y tamaño, haciendo que la instalación de redes de sensores inalámbricas sean actualmente llamativas en diferentes áreas.

El concepto de redes de sensores inalámbricos se basa en una ecuación simple: Detección + CPU + Radio = Miles de aplicaciones potenciales [27]. Lo que significa que los sensores miden las condiciones del entorno y luego esta información sea transformada y procesada, para finalmente transmitirse, mostrando algunas características acerca de algún fenómeno alrededor de estos sensores, dando como resultado que las aplicaciones de WSNs (Wireless Sensor Networks) sean bastante numerosas. Por ejemplo, en el monitoreo del medio ambiente, agricultura, seguridad, aplicaciones miliares, medicina y para este caso en las microrredes.

Son tan importantes los avances que se pueden llegar a tener con WSNs, que las empresas de proyectos de investigación, institutos de investigación y universidades, han estado desarrollando dispositivos para WSNs , como la "ZigBee Alliance" una asociación de empresas que trabajan en el desarrollo de dispositivos de control y monitoreo, la cual anunció en el año 2004 la publicación de la norma ZigBee, para redes de sensores inalámbricos, basada en el estándar IEEE 802.15.4 [26].

3.1 ELEMENTOS DE UNA RED DE SENSORES INALÁMBRICA.

Las redes de sensores inalámbricos están conformadas por nodos o también conocidos como motas, quienes tienen la capacidad de interactuar con el medio ambiente mediante para la detección o el control de parámetros físicos. Estos nodos están conectados inalámbricamente y son capaces de comunicarse y colaborar entre sí [28]. En la figura 10 se ilustra el despliegue de los elementos de una WSN.

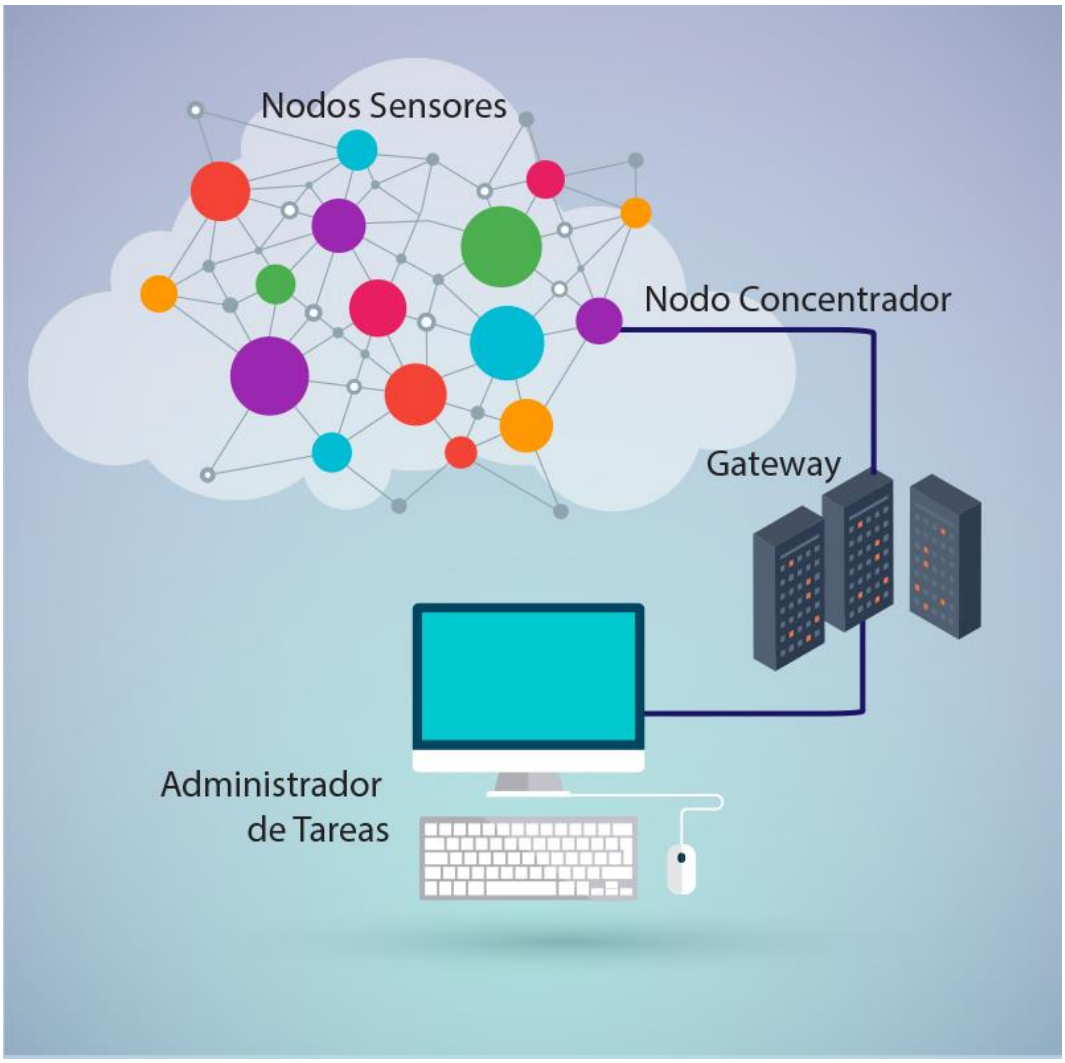


Figura 10 Despliegue clásico de un WSN inteligente (adaptación de la revista IEEE Wireless Communications 2004) [14].

- Nodos sensores/actuadores

El nodo sensor detecta los eventos, realiza un tratamiento de datos local, transmite y retrasmite la información obtenida.

- Nodo concentrador/controlador.

El nodo concentrador recibe los datos de los demás nodos sensores, realiza un tratamiento y los envía hacia el sistema de gestión de la información. Este tiene que hacer un diagnóstico del estado de los nodos sensores y en caso de fallos aplicar mecanismos de recuperación y contingencia.

- Gateway

Son los elementos destinados a la interconexión entre la red de sensores y una red de datos. Los Gateway pueden clasificarse como activo, pasivo e híbrido. Un gateway activo permite que los nodos sensores envíen directamente sus datos al Gateway Server, mientas que un gateway pasivo funciona mediante el envío de una solicitud al nodo sensor. Finalmente el gateway híbrido combina las capacidades de los activos y pasivos [29].

- Administrador de tareas

Toda la información de datos procedentes de los nodos sensores se almacena en los gestores de tareas para el análisis, donde los usuarios pueden analizar esta información de forma local o remota.

3.2 ARQUITECTURA DEL NODO SENSOR

Un nodo sensor básico está típicamente formado por cuatro componentes principales; el módulo sensor, un módulo de procesamiento, el módulo de comunicaciones y un módulo de energía como se presenta en la figura 11. Una de las consideraciones de diseño más importante es la eficiencia del consumo de energía por lo que los componentes entrelazados tienen que operar y equilibrar con el menor consumo de energía y cumplir todas sus funciones.

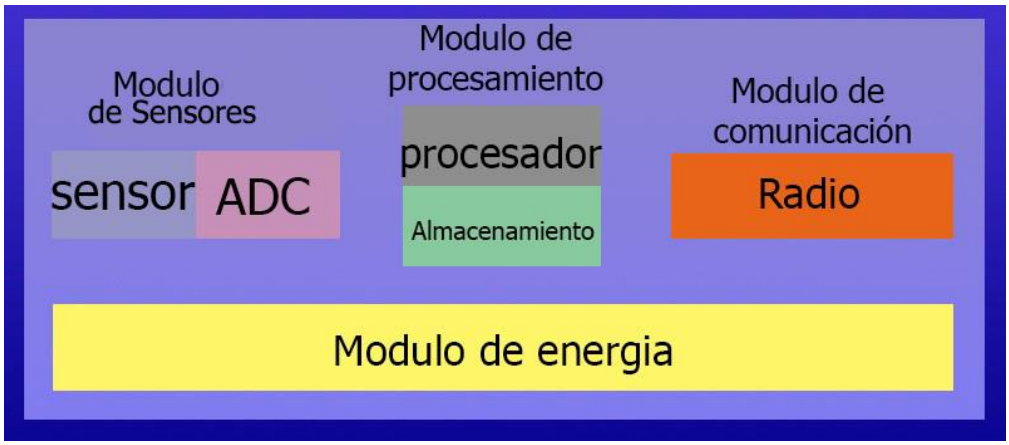


Figura 11 Componentes de un nodo sensor (adaptación de la revista IEEE Wireless Communications 2004) [30].

Componentes del nodo sensor

- Módulo de Sensores: se compone del sensor y ADC, el sensor detecta y captura la variable a medir, a través del ADC se digitaliza para enviarse al módulo de procesamiento, también le indica al módulo de sensores qué hacer.

- Módulo de Comunicación: Suele usarse radio frecuencia, ya que proporciona largo alcance y velocidades altas de transmisión, también tazas de error permisibles y un bajo uso de energía. Sin embargo, dependiendo de la aplicación puede usarse dispositivos de ultrasonido, óptica o inductancia magnética.
- Módulo de Procesamiento: Interpreta la consulta del ADC, monitorea y controla la distribución de energía, realiza el procesamiento de los datos recibidos, toma decisiones de envío de datos, ejecuta protocolos y programas de aplicación. En este módulo se puede hacer uso de procesadores convencionales como los de los computadores, pero dado al consumo de energía limitado y capacidad de almacenamiento requerida, no es factible, por lo que se puede usarse microcontroladores y procesamiento digital de señales DSPs (Digital Signal Processors), circuitos integrados de aplicación específica ASIC (Aplication-Specific Integrated Circuit) o dispositivos de lógica programable FPGAs (Fiel Programable Gate Array). Dependiendo de la aplicación es importante definir el tipo de unidad de procesamiento, ya que va a estar limitada por los recursos energéticos, capacidad y velocidad de procesamiento, costos, ambiente en el que se encuentre el nodo sensor, tipo de datos que se van a manejar, capacidad de almacenamiento, multiprocesamiento, manejo de interrupciones y asignación de memoria [26].

3.3 SISTEMA DE COMUNICACIÓN.

Los sistemas de comunicación que se utilizan en las redes de sensores para microrredes varían según en las especificaciones técnicas que tienen cada una de las tecnologías, donde el factor que determina la selección de los dispositivos de comunicación son las características físicas y elementos que pertenezcan a la microrred.

En la figura 12 se identifican el tipo de red a la que pertenecen de acuerdo a la ubicación de los usuarios, según las características de información o las necesidad de comunicación que se requiere en ese punto de la microrred y de esta forma asignar los dispositivos más adecuados para el transporte de la información sensada a través de la red de comunicaciones [31], [32].

Un factor adicional en la red de sensores es la interacción del usuario, de la manera que no puede realizar las tareas de control pero puede saber con precisión el consumo de energía casi inmediata de realizar y tomar decisiones sobre el uso de la electricidad [33], [34], [35].

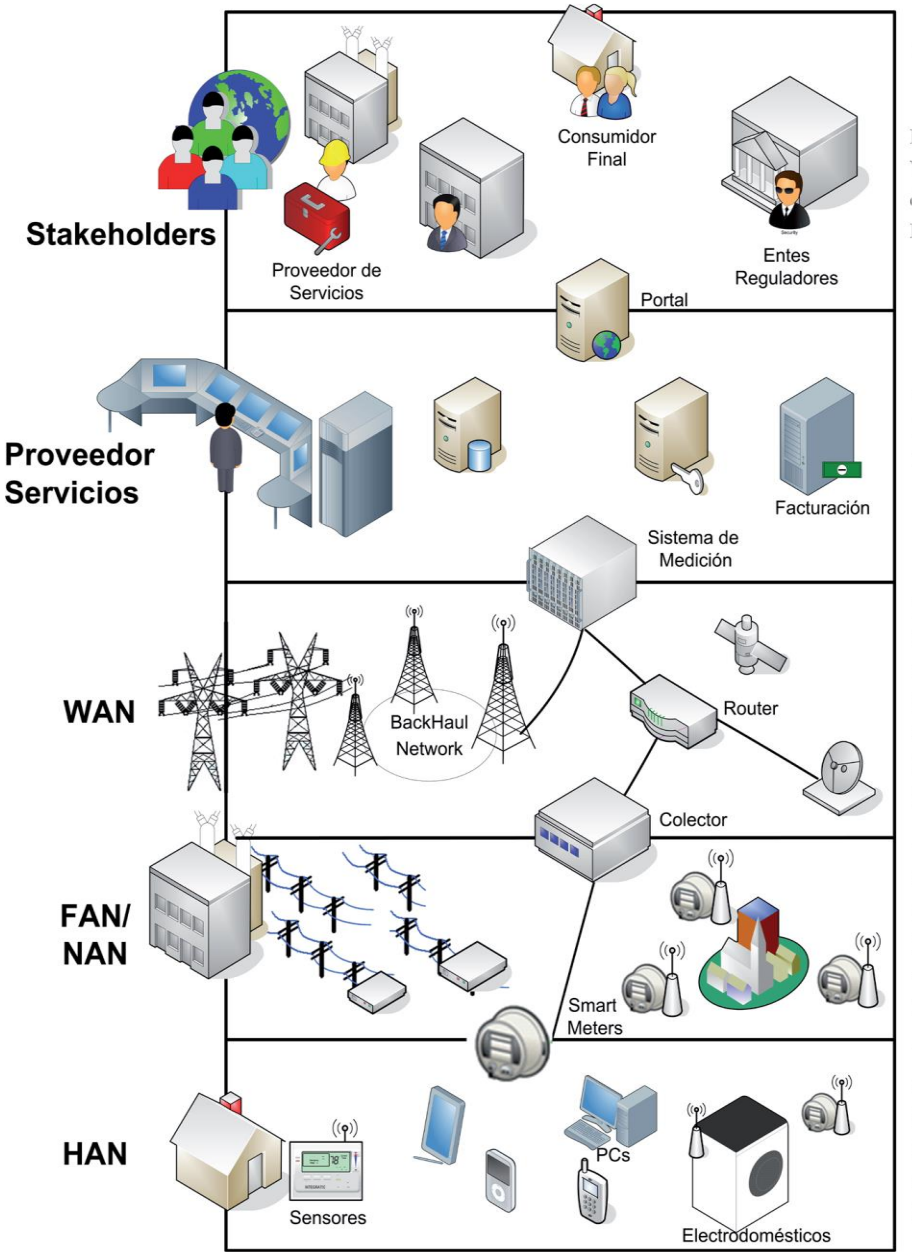


Figura 12 Tecnologías de comunicación para una microrred acorde a la compatibilidad de los dispositivos [9].

Conociendo el nivel de jerarquía en el cual se encuentra el punto donde se desea realizar la transferencia de datos en la microrred, se asocia esa arquitectura de comunicación a una tecnología y/o a un protocolo en específico. En la tabla 1 se establece la relación de Tecnologías y protocolos de comunicación con la arquitectura de comunicación, se considera a manera de guía y no una selección estricta.

Arquitectura de comunicación	Tecnologías y protocolos de comunicación
Usuarios y proveedores de Servicio	Protocolos de Internet WWW (World wide web) ebXML IEC60870
Proveedores de Servicio	Web IEC 61970 IEC 61968
WAN	IP/MPSL SONET,WDM,ATM EVDO/EDGE/HSDPA 802.16/LMDS GPON/EPON DOCSIS Metro-Ehternet
FAN/NAN	Wimax 802.16 802.11 n/g 802.15.4/ZigBee DOCSIS xDSL PLC/BPL
HAN	802.15.4/ZigBee 802.11 n/g Z-wave 6LowPAN HomePlung

Tabla 1 Tecnologías y protocolos de comunicación [36] [31],[37],[38] .

3.3.1 Tecnologías inalámbricas

- WI-FI

Esta tecnología de comunicación inalámbrica está basada en el estándar IEEE 802.11 o WLAN. Se pueden encontrar dos tipos de comunicación en WI-FI, una a una tasa de 11Mbps y otra a 54 Mbps, con un ancho de banda de 20 MHz puede tener un alcance de 100 a 150 metros, lo que la hace la más atractiva para acceder a internet inalámbricamente.

- WIMAX

Esta tecnología permite la recepción de datos por microondas y la retransmisión por ondas de radio, se basa en el estándar IEEE 802.16. Esta tecnología inalámbrica está diseñada para

una red de área metropolitana con cobertura de 50 km y con tazas de transmisión hasta 70 Mbps.

- Bluetooth

Se basa bajo la especificación industrial IEEE 802.15.1 para redes inalámbricas de área personal, está diseñada para establecer una comunicación inalámbrica entre varios dispositivos en la banda de frecuencias 2.4 GHz. Estos dispositivos se clasifican como:

Clase 1: que tiene una potencia máxima de 100 mW y un alcance de 100 metros.

Clase 2: que tiene una potencia máxima de 2.5 mW y un alcance de 10 metros.

Clase 3: que tiene una potencia máxima de 1 mW y un alcance de 1 metro.

- Zigbee

Establece como principal objetivo el de comunicar aplicaciones que requieren una comunicación segura, con tasas bajas de envío y bajo consumo. Se basa en dispositivos inalámbricos operando en la banda ISM para usos industriales, científicos y médicos (868 MHz, 915 MHz y 2.4 GHz) con una modulación en espectro ensanchado por secuencia directa (DSSS) también conocida como acceso múltiple por división de código en secuencia directa (DS-CDMA). En el rango de frecuencias de 2.4 GHz (banda más extendida) se definen 16 canales con un ancho de banda de 5 MHz [39]. Se basa en el estándar 802.15.4. Estas redes se caracterizan por su simplicidad y bajo costo, proporcionando conectividad inalámbrica en aplicaciones que requieren un consumo de potencia muy reducido [40]

- IEEE 802.15.4

Es un estándar que define el nivel físico y el control de acceso al medio de redes inalámbricas de área personal con tasas bajas de transmisión de datos (low-rate wireless personal área network, LR-WPAN). En su forma básica se concibe en un área de comunicación de 10 metros con una tasa de transferencia de 250 kbps [41].

3.4 SISTEMAS DE MEDICIÓN INTELIGENTE O AVANZADA.

La creciente demanda de electricidad ha acelerado la necesidad de modernizar la red distribución, y con ayuda de las tecnologías que se han desarrollado se generan los sistemas de medición inteligente.

Los sistemas de medición inteligentes permiten el intercambio de información entre el usuario y el proveedor, para la ayuda en la gestión del servicio, así mismo permiten tener control y evitar fallas en la red eléctrica. Esto se logra usando medidores inteligentes, ya que su desarrollo se debe a los adelantos en los dispositivos electrónicos y las tecnologías de las telecomunicaciones con mayores capacidades de registro y transferencia de datos en comparación con los medidores electromagnéticos convencionales. Los primero pasos hacia

estos sistemas han sido los sistemas AMR (Lectura Automática de Medición) y los AMI (Infraestructura de medición Avanzada) [42].

3.4.1 Lectura Automática de Medición AMR.

Este tipo de sistemas implementaron módulos de comunicación, lo cual permite realizar una lectura del consumo de electricidad de manera remota sin necesidad de inspeccionar el dispositivo de manera directa como se realiza con los medidores tradicionales, esto permite tener la información del consumo en todo momento [43].

La información recolectada se envía a un centro de control donde la información se almacena y gestiona, la comunicación que se realiza en este sistema es unidireccional, esto no permite que el centro de gestión envié señales al medidor, por lo que se conoce como lector remoto.

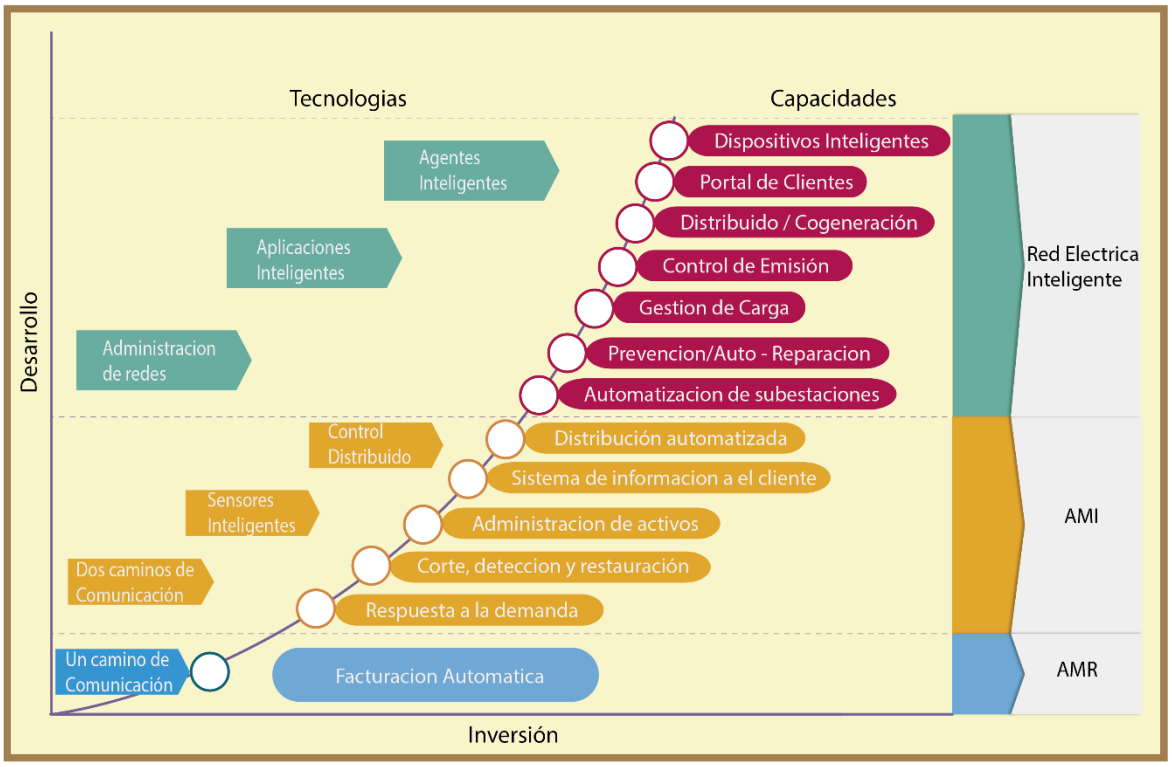


Figura 13 Jerarquía de una red inteligente (adaptación de la revista IEEE power & energy magazine 2010) [30].

Como muestra la figura 13, aunque la tecnología AMR demostró ser inicialmente atractiva, AMR no aborda la cuestión más importante por resolver: la gestión de la demanda. Debido a su comunicación unidireccional del sistema, la capacidad de AMR está limitada [30].

En consecuencia, la tecnología AMR no tiene las características suficientes para tener un sistema de medición inteligente, pero si presenta un gran avance. Los sistemas AMR, evolucionaron hacia los sistemas la infraestructura de medición avanzada [44].

3.4.2 Infraestructura de medición Avanzada (AMI).

AMI además de poseer todas las características de los sistemas AMR, plantea adicionalmente enlaces de comunicación bidireccional permitiendo el intercambio de información entre un punto de control y el medidor, de esta forma se logra efectuar una gestión con los datos de lectura en tiempo real, proporcionando información acerca del consumo energético entre otros parámetros que puedan interesar a el nodo central para su análisis [45].

Este sistema incluye el hardware y el software de comunicaciones, el sistema asociado y el software de gestión de datos en un nodo central, que crea una red entre distintos dispositivos de medición avanzada permitiendo la recolección y distribución de información [46].

1. Características y funciones del sistema AMI.

Los sistemas de medición inteligente permiten registrar y analizar diferentes parámetros, según las características o tipos de sensores que lo acompañen, como los son agua, gas, vapor y energía eléctrica, teniendo en este último una mayor presencia [7].

Las funciones que tiene AMI en las microrredes son muy extensas, permite a los usuarios acceder a los datos de consumo de manera completa en tiempo real o en determinados intervalos de tiempo con un nivel de precisión alto, de igual manera los datos adquiridos pueden transmitirse por medio de una red de comunicaciones a un nodo central de análisis. De esta manera se crea un flujo de información entre todos los dispositivos que hacen partes de la microrred, la información adquirida permite desarrollar pronósticos de demanda, cobertura, gestión de la oferta, disminuir la cantidad de fallos que se puedan presentar en la microrred ya sea en los sistemas de distribución o en los puntos de generación, optimizando así el uso de la energía, reduciendo costos y controlando los recursos [10].

La función de comunicación bidireccional, permite realizar lecturas bajo demanda, detectar manipulaciones y condiciones de tensión fuera de rango, adicionalmente es posible realizar una reacción más rápida y precisa al momento de presentarse una falla y así proceder el restablecimiento del servicio.

Las características más relevantes que debe tener este sistema son:

- Comunicación Bidireccional.
- Fecha y Hora exacta de la medición realizada.
- Capacidad de detectar, reconfigurar y reparar problemas de comunicación, para garantizar la disponibilidad de las mediciones.
- Capacidad de lectura remota de medidores bajo demanda, es decir, cuando le sea solicitada dicha información.
- Capacidad de actuar en la red si le es notificado
- Precisión en los datos adquiridos de la red.
- Informe sobre detección de fallas.

Hay otras características que pueden incluirse en AMI, dependiendo de la aplicación o los requerimientos que se quieran suplir.

Un aspecto muy importante es el nivel de seguridad de los módulos de trasmisión de datos, de esta forma se evitan sabotajes o ataques, con este propósitos se han desarrollados protocolos y métodos de encriptación que aseguran la transmisión segura de información, que según los diferentes sistemas de comunicación varían entre si proporcionando la seguridad en el proceso [12], [13].

3.5 COMPONENTES DEL AMI

Los componentes básicos de AMI se pueden observar en la figura 14. El usuario dispone de un equipo que será un medidor electrónico avanzado que recopilan datos en el tiempo. Las mediciones incluyen diferentes parámetros de la red eléctrica según se requiera. Estos medidores tienen la capacidad de transmitir los datos, que son recibidos por el sistema de central que realiza la gestión [45].

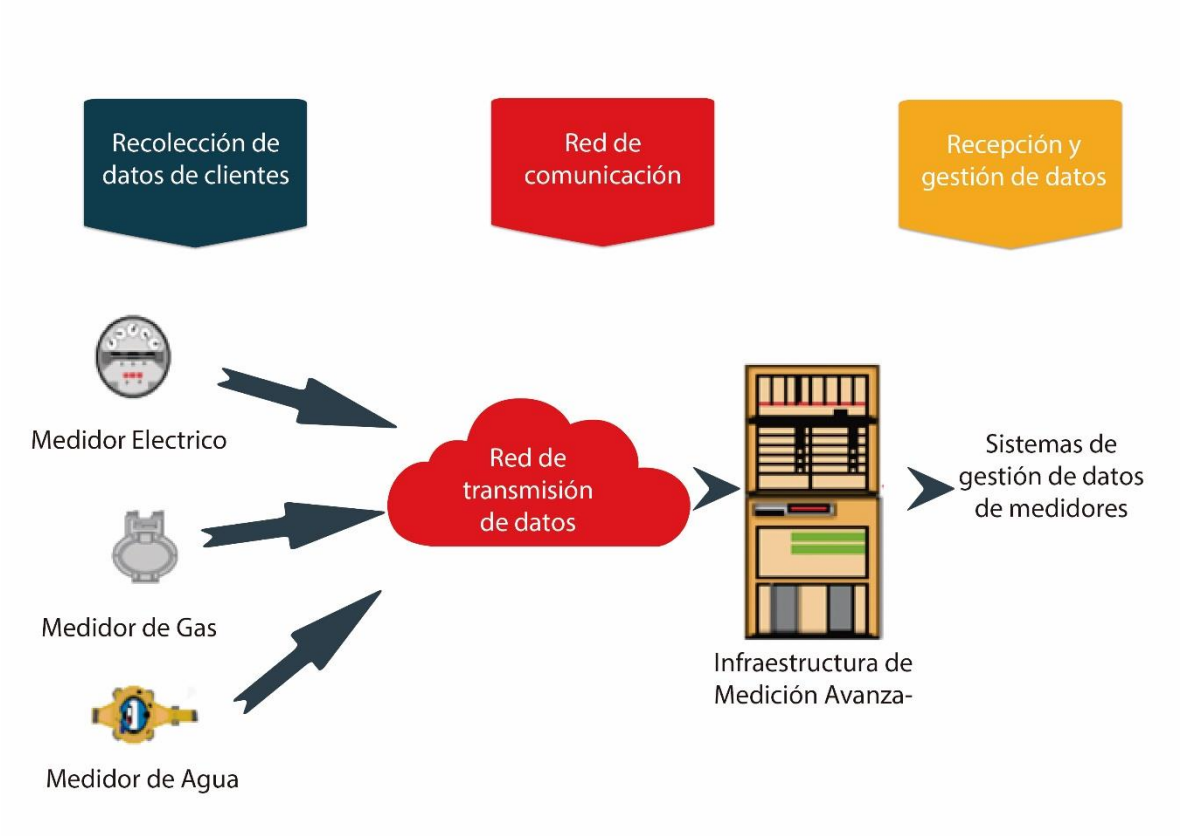


Figura 14 Jerarquía de una red inteligente (adaptación de la revista Electric Power Research Institute 2007) [45].

Un medidor avanzado registra los datos de manera constante y los envía a través de una red de comunicación, el medidor inteligente habilita dos vías de comunicación entre el medidor y el sistema central, el objetivo fundamental del medidor eléctrico, es registrar de manera

precisa el consumo energético, realiza el registro de la medición y almacenamiento, verificando el estado del servicio, para luego enviar al concentrador o sistema de gestión.

Está conformado por una serie de sensores que adquieren la información que se considere importante de la red eléctrica, estas señales sensadas se acoplan para posteriormente procesarlas y enviarla a través del sistema de comunicación. Esto se realiza mediante un microcontrolador que procesa la información medida. También son usados los sistemas de medición como interruptores inteligentes [11].

El modelo básico de un sistema AMI está compuesto por, un módulo de sensores, un módulo de conversión análoga digital (ADC), procesamiento de datos y un módulo de comunicación receptor transmisor como se observa en la figura 11.

3.5.1 Conversor análogo digital ADC.

Un dispositivo Conversor análogo digital o ADC, convierte señales analógicas en valores digitales o binarios, Los avances en el procesamiento digital de las últimas décadas motivan el hecho de poder reducir señales continuas a muestras discretas. Esto con el fin de poder hacer un procesamiento de las señales adquiridas de manera más simple. Los Convertidores son componentes esenciales que proporcionan un vínculo entre digital y analógico y son empleados en Redes ópticas, sistemas de comunicación móvil, imágenes médicas, reproductores de sonido, de vídeo, que son sólo unos pocos ejemplos de las aplicaciones que requieren un ADC.

La cantidad de bits en las que puede ser representada una señal analógica determina, la resolución del paquete de información, banda de error, es decir si se tiene una gran cantidad de bits que representan una señal en un instante determinado se tiene un valor más preciso de la conversión que se está realizando, la consideración consiguiente a esto es la determinación de cuantos bits ofrecen una conversión optima y fiable de la señal análoga.

Para la transformación de una señal en tiempo continuo, en tiempo discreto se hace necesario realizar un muestreo de la señal, para esto se toman muestras a una determinada frecuencia del valor de la señal continua, creando un intervalo de muestreo de un periodo, determinándose por consiguiente los intervalos de adquisición de datos que también puede ser conocida como frecuencia de muestreo.

Hay diferentes tipos de convertidores, cada uno difiere de las propiedades o de la aplicación en la cual se desea implementar, siendo los parámetros más importantes para la selección del mismo los siguientes:

- Resolución de bits.
- Tiempo de conversión.
- Características dinámicas y estáticas.
- Consumo de energía.

- Precio.

Las arquitecturas más comunes para conversión de señales son, Flash o paralelo, Aproximaciones sucesivas, rampa simple, doble rampa, entre otras, donde los parámetros mencionados anteriormente las hacen únicas y viables según la aplicación en la cual se deseen utilizar [47].

3.5.2 Sistemas de comunicaciones

La red de comunicaciones, es el medio para el intercambio de información entre el punto de adquisición de datos y el centro de monitoreo, La comunicación bidireccional sin perdidas de datos es el aspecto clave, para esto se usa tecnologías alambradas e inalámbricas disponibles, los medios inalámbricos permiten una comunicación fiable entre los dispositivos y suelen ser los más utilizados [10].

La infraestructura de comunicación puede contener varios medios y tecnologías para la transmisión de información, como lo puede ser por redes inalámbricas, microondas, PLC, fibra óptica entre otras, permitiendo flexibilidad, escalabilidad y que pueda tener un sistema de comunicación hibrida.

Existe una dependencia de las características del sitio donde se presta el servicio, puesto que los problemas de comunicación en zonas rurales son diferentes a los que se puedan presentar en las zonas urbanas, o la disposición geográfica del terreno también influye en la tecnología de comunicación que se selecciona.

3.5.3 Red Inalámbrica

Las redes inalámbricas usan ondas electromagnéticas para en enlazar nodos de comunicación, estas redes de radio frecuencia emplean diferentes tecnologías como Wi-Fi, Zigbee, Wi-MAX, líneas telefónicas, entre otras, hasta llegar al nodo recolector. Para evitar la interferencia que se puede presentar por los dispositivos actuales que utilizan esta tecnología se deben considerar las normas o estándares, para esto existen estándares inalámbricos a utilizarse en los medidores inteligentes como el estándar IEEE 802.11 para redes de área local (LAN), el estándar IEEE 802.16 basada World wide para Microondas (Wi-MAX), redes de telefonía celular (3G y 4G), ZigBee [10] basado en el estándar IEEE 802.15.

3.5.4 Red PLC

Es la tecnología que utiliza las líneas eléctricas para transmitir señales de radio habilitando la comunicación de datos, convirtiendo la línea tradicional de distribución en una línea digital, presenta problemas cuando se manejan volúmenes de datos de gran magnitud pero se tiene una confiabilidad alta en la información transmitida por los medidores, por sus características es un método de comunicación ideal cuando se necesitan realizar mediciones en entornos rurales.

3.5.5 Red Hibrida

Las redes hibridas son una combinación de todas las tecnologías de comunicación que existen, operan donde una sola arquitectura de comunicación no es viable para la implementación, aprovecha las características especiales de cada estructura y las emplea de manera principal según las disposiciones geográficas, en la figura 15 se observa la integración de tecnologías de comunicación de radio frecuencia con tecnologías cableadas como la fibra óptica, que se unen en una oficina central de recolección conectada con otros puntos de la red [34] .

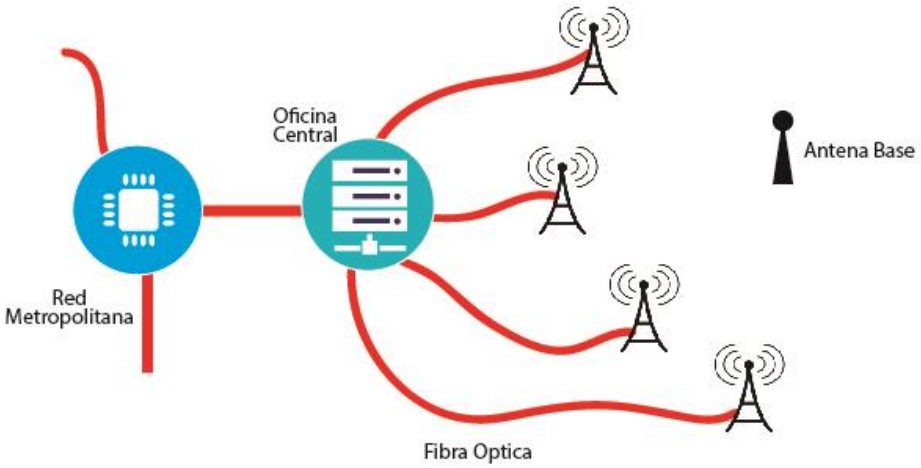


Figura 15 Esquema de red hibrida (adaptación de revista Óptica Pura y Aplicada) [34].

3.5.6 Gestión

El sistema de gestión de datos medidos o MDM (Meter Data Managent) ,toma la información que ha sido recolectada desde los medidores inteligentes al concentrador de datos y transmitida a través de la infraestructura de telecomunicaciones al punto central, necesita que sea tratada en un sistema de gestión, esto con el fin de lograr el propósito de una red AMI, este componente requiere de un software de gestión., el MDM dispone de las capacidades de monitorear y controlar la información, automatizando el proceso permitiendo actuar con acciones de corte y reconexión [42].

Este sistema debe ser capaz de tener un óptimo rendimiento, ser confiable y mantener el sistema actualizado en todo momento sin tiempo de inactividad; para mantener la información disponible, debe soportar eventos o desastres del sistema principal, así que debe tener un respaldo de recuperación con un nivel de seguridad robusto que puede ser por permisos de nivel de usuario como leer o editar e intrusión externa indebida. En la figura 16, se puede observar un esquema general de la red haciendo especial énfasis en las labores de gestión que se quieren suplir [48], [42].

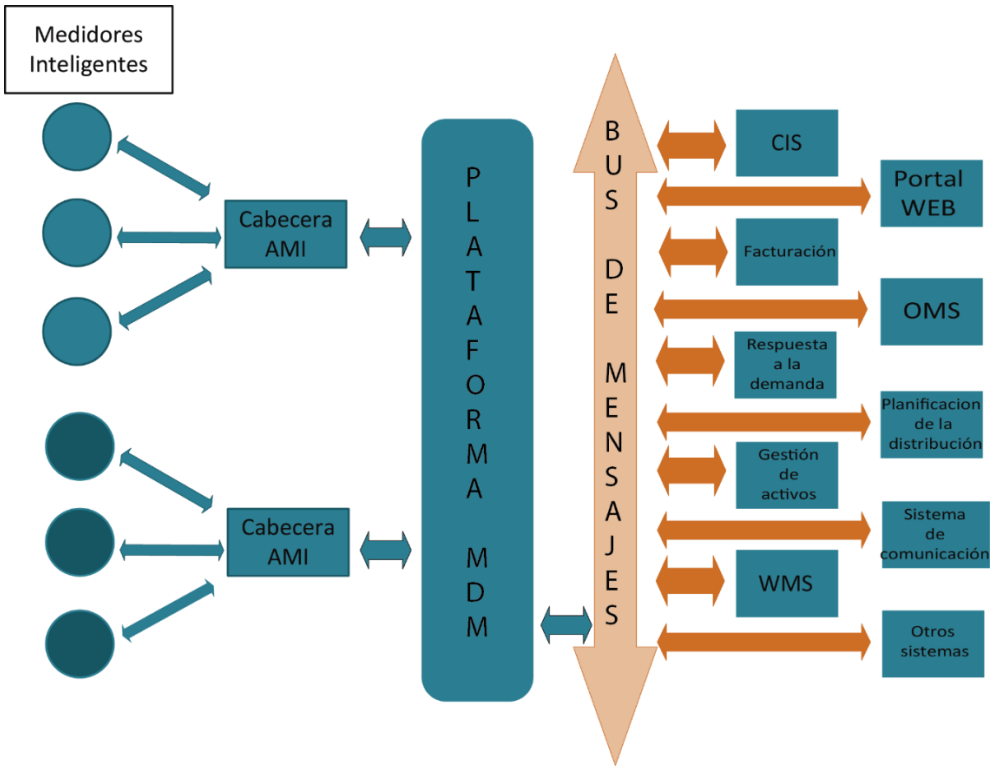


Figura 16 Esquema de red para labores de gestión(Adaptación) [49].

4
DISEÑO DE LA RED DE SENSORES INALÁMBRICO

Es necesario identificar las distintas tareas que deben cumplir la red de sensores, a partir de esta información se establecen las etapas que permiten realizar el diseño de la misma, en la figura 17 se establece el procedimiento que se llevara a cabo para el desarrollo de esta red.

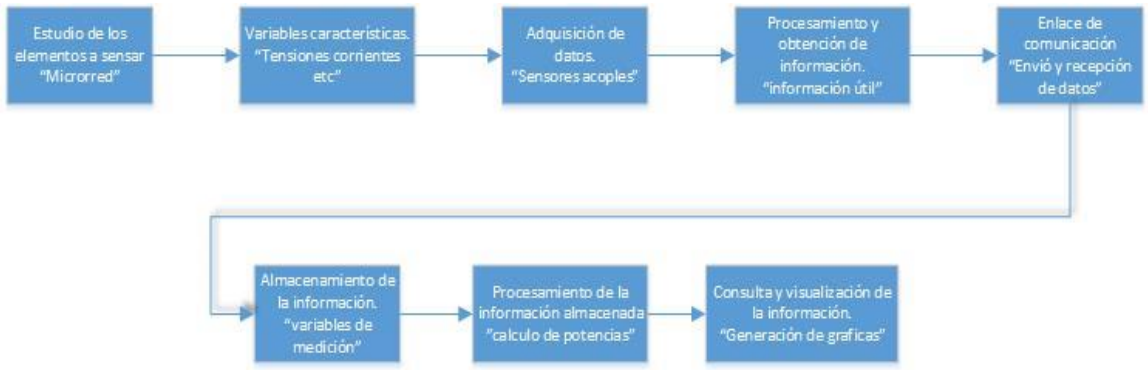


Figura 17 Diagrama para el diseño de una red de sensores de una microrred

4.1.CRITERIOS DE DISEÑO DEL NODO SENSOR

Para el diseño del nodo sensor se debe asegurar que pueda operar de manera correcta con las desviaciones de los valores establecidos para las variables de tensión y la forma de las ondas de tensión, corriente y frecuencia. Establecidas por la CREG en la Resolución 097 de 2008, el cual es el ente encargado de regular los servicios públicos domiciliarios de energía eléctrica, gas licuado de petróleo y combustibles líquidos en Colombia [50], [51].

Dependiendo de la aplicación para la que se diseñe la red de sensores inalámbricos se deben tener unos requisitos. A continuación, se enuncian los requisitos y objetivos principales para una WSN en general:

1. Nodo Pequeño: Un tamaño pequeño facilita el despliegue de la red, así mismo reduce el consumo de energía y costo de los sensores. Los nodos sensores generalmente se despliegan en un entorno hostil en grandes cantidades sin la posibilidad de reutilizarse, reduciendo el coste de los nodos se reduce el coste de toda la red.

2. Bajo consumo de energía: El consumo de energía es crucial para que la red funcione continuamente, al usar baterías o un tipo de alimentación alternativa. Es importante tener en cuenta el consumo que puede llegar a tener cada módulo del sensor ya que permite prolongar la vida del sensor.
3. Escalabilidad: Se debe elegir un protocolo de encaminamiento para que la red sea escalable.
4. Confiabilidad: Los protocolos de la red de sensores deben proporcionar los mecanismos de control y corrección de errores para garantizar la entrega de datos fiable a través de canales inalámbricos ruidosos, propensos a errores, y variables en el tiempo.
5. Auto-configurabilidad: Una vez desplegada la red de sensores deben ser capaces de organizar autónomamente a sí mismos en una red de comunicación y reconfigurar su conectividad en caso de cambios en la topología y fallos en los nodos.
6. Adaptabilidad: Los protocolos de red de sensores deben ser adaptables a los cambios de densidad y topología.
7. Uso del canal: Las redes de sensores tienen recursos de ancho de banda limitado, por lo que protocolos de comunicación deben hacer eficiente el uso del ancho de banda para mejorar el uso del canal.
8. Tolerancia a fallos: Los nodos sensores deben ser tolerante a fallos y tener las habilidades de auto-prueba, auto-calibración, auto-reparación y auto-recuperación ya que los nodos son propensos a fallas debido a los entornos de despliegue y operaciones desatendidas.
9. Seguridad: Es necesario introducir mecanismos de seguridad eficaces para prevenir el acceso no autorizado a ataques maliciosos a la información que se transmite sobre la red o en los nodos.

4.2.VARIABLES A MEDIR.

Identificar y conocer cuáles son las variables necesarias para el análisis de la red es la parte fundamental para la selección de elementos. El dispositivo de sensado debe ser capaz de suministrar toda la información que se considere relevante de la red de distribución eléctrica y así realizar un análisis de la energía, rendimiento o potencia del sistema.

La potencia ya sea la que se genera o la que se consume, suele ser la variable con mayor interés, para conocer el valor de esta variable se parte de las ecuaciones mencionadas en el análisis de potencia presentado anteriormente.

Este dispositivo es de carácter bidireccional, lo cual permite conocer las variables deseadas en cualquier punto de la microrred, sin importar la topología y los elementos que componen la microrred, estos se ubicarán en dos tipos de elementos y se considerarán como un nodo y clasificarán así:

- Generadores o aportadores de energía
- Cargas o consumidores de energía.

Cuando se habla de generación, variables como la frecuencia y fase de la red, aparte de las ya mencionadas tienen una gran importancia debido a que los convertidores tienen la tarea de realizar la sincronización con la red asegurando la frecuencia y la fase de la señal, monitorear estas variables permite cumplir con los estándares de calidad de generación de energía; Estos estándares obligan a mantener ciertas características de la señal generada, de no cumplirse puede afectar a todos los dispositivos de la red, en especial las cargas o los usuarios puesto que ellos no poseen medidas de protección, confiando solamente en la calidad del servicio a su proveedor o sistema de generación, esto permite tener un mayor un conocimiento del sistema.

Las variables necesarias para realizar un análisis detallado del sistema son:

- Tensión de línea.
- Corrientes de línea.
- Factor de potencia.
- Frecuencia.

Los elementos de medición empleados para el desarrollo de esta investigación, permiten a partir de su principio de funcionamiento extraer otras variables que son necesarias en el análisis detallado de la microrred, la obtención de estas variables y el principio de funcionamiento de los sensores se explicara de manera más detallada en las siguientes secciones.

4.3. ELEMENTOS DE MEDICIÓN Y PRINCIPIO DE FUNCIONAMIENTO.

Los sensores entregan una señal o una reacción, que está en función de una señal de entrada que está aprovechando algún fenómeno físico, en este caso se opta por el uso de sensores de bucle cerrado o con aislamiento galvánico (CL), teniendo la ventaja de aislar el dispositivo de medición de la red eléctrica.

- Transductores de corriente CL

  El flujo magnético creado por la corriente primaria genera un flujo proporcional en el bobinado secundario. El sensor de efecto Hall y la electrónica asociada se encarga de generar la intensidad de compensación, que es una representación exacta de la corriente primaria, el esquema general se puede aprecian e la figura 18, donde *IN* es la señal de entrada, la polarización del dispositivo son las indicadas como *0V* y *5V*, *OUT* es la señal que entrega y *Ref* es un valor de referencia externo que se puede aplicar a la señal para añadir un nivel DC a la señal, por defecto el sensor entrega este nivel en 2.5V.

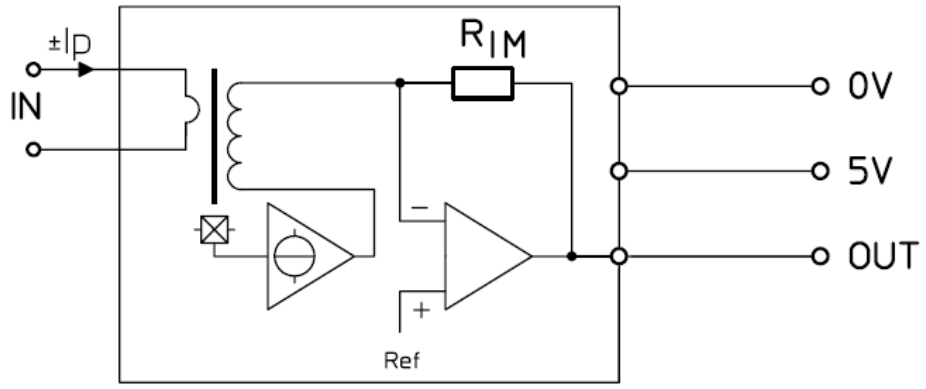


Figura 18 Principio de operación transductor de corriente, tomado del fabricante LEM.

- Transductores de Tensión CL

En la figura 19 se muestra el esquema general del sensor de tensión. La tensión se toma del bobinado principal a partir de una pequeña corriente limitada por una resistencia en serie nombrada *R1*, El flujo magnético creado por la corriente del primario genera un flujo complementario mediante el devanado secundario. Junto con el circuito electrónico asociado se genera en el secundario la corriente, que es una representación exacta de la tensión del primario nombrada *Is*. Es necesario introducir una resistencia de carga de forma externa para convertir la corriente generada en el bobinado secundario en un valor de tensión que se nombra como *RM*, esto permite de igual manera ajustar o configurar el rango de medición del dispositivo. *+Uc* y *–Uc* hacen referencia a los valores de polarización. A diferencia del transductor de corriente que nos entrega una señal DC, este elemento nos entrega una señal AC.

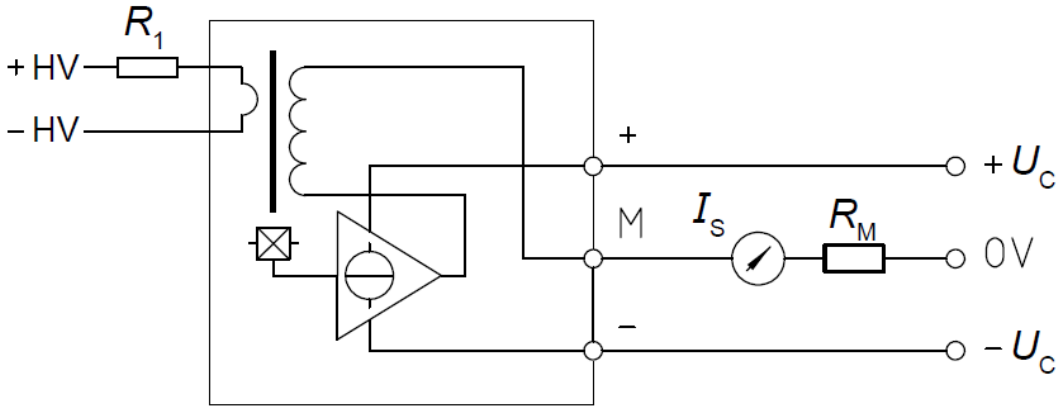


Figura 19 Principio básico de funcionamiento, circuito equivalente (de izquierda a derecha).

Al emplear dispositivos con este principio de funcionamiento se permite:

- Amplio ancho de banda.

- Buena precisión global.
- Rápido tiempo de respuesta.
- Excelente linealidad.
- Medición de altas tensiones.
- Aislamiento de seguridad
- Baja temperatura

Los dispositivos seleccionados para el sensado pertenecen a la marca LEM, estos ofrecen una alta calidad en sus dispositivos y cumplen diferentes estándares internacionales de operación, en la tabla 2 y 3 se presentan las características más relevantes de estos elementos.

- Características sensor de tensión (LV-25 P), tabla 2.

Característica	Valor
Tipo de sensor	Efecto Hall, lazo cerrado
Para medir	AC/DC
Corriente primaria nominal	± 14mA
Rango de medición	500V
Salida	Tensión
Conversión ratio	2500:1000
Corriente nominal del segundario	25 mA
Frecuencia	DC - 200 kHz
Linealidad	± 0,1%
Exactitud	± 0,9%
Suministro de tensión	± 12V ± 15V
Tiempo de respuesta	0.4μs
Temperatura de funcionamiento	-40 ° C ~ 85 ° C
Polarización	bidireccional

Tabla 2 Sensor LV-25 P LEM.

- Características sensor de corriente (LTSR-25), tabla 3.

Característica	Valor
Tipo de sensor	Efecto Hall, lazo cerrado
Para medir	AC/DC
Corriente primaria nominal RMS	25A
Rango de medición	± 80A
Salida	Tensión
Sensibilidad	25mV / A
Frecuencia	DC - 200 kHz
Linealidad	± 0,1%
Exactitud	± 0,2%
Suministro de tensión	5V

Tiempo de respuesta	0.4µs
Temperatura de funcionamiento	-40 ° C ~ 85 ° C
Polarización	bidireccional

Tabla 3 Sensor LTSR-25 LEM.

4.4.TOPOLOGÍA DE MEDICIÓN.

Conociendo los elementos que se disponen para el sensado con su principio de operación, junto con las variables necesarias para conocer el comportamiento de la red, se puede realizar un diagrama de distribución de los elementos de sensado dentro de la red eléctrica trifásica. En la figura 20, se observa la necesidad de usar tres sensores de tensión y cuatro sensores de corriente.

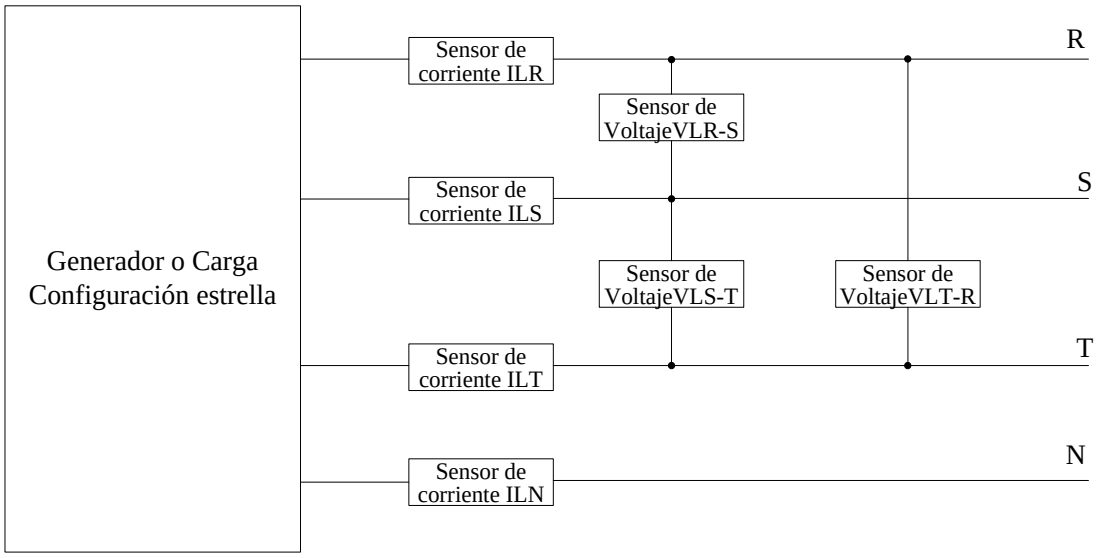


Figura 20 Disposición de sensores en la red trifásica.

El esquema eléctrico de la disposición de los sensores se presenta en el anexo 11.1.

4.5. PROCESADOR DIGITAL DE SEÑAL (DSP).

Para el desarrollo del proyecto se contó con la plataforma de desarrollo TMS320F28335 Experimenter Kit de Texas Instruments. El cual consta de la tarjeta de control donde se encuentra el DSP y la base de acoplamiento de la familia Delfino C2000, en la figura 21 se puede observar el programador y en la figura 22 la tarjeta de procesamiento.

Las principales características del procesador son las siguientes:

- Tecnología CMOS de alto desempeño,
  - Frecuencia de nucleó hasta 150MHz (6.67 ns ciclos de instrucción).

- Tensión de alimentación de 1.8V para dispositivos flash, 1.9V para RAM/ROM.
- 3.3V (entrada y salida) en puertos de propósito general.

- CPU de 32 bits de alto desempeño.
  - Unidad punto flotante estándar IEEE-754.
  - 16 x 16 y 32 x 32 Operaciones MAC
  - Arquitectura de buses Harvard.
  - Respuesta rápida a interrupción y procesamiento.
  - Código eficiente (en C / C ++ y Assembler).
- 256 KB × 16 de memoria FLASH y 34 KB × 16 de memoria SARAM.
- Operación en punto flotante.

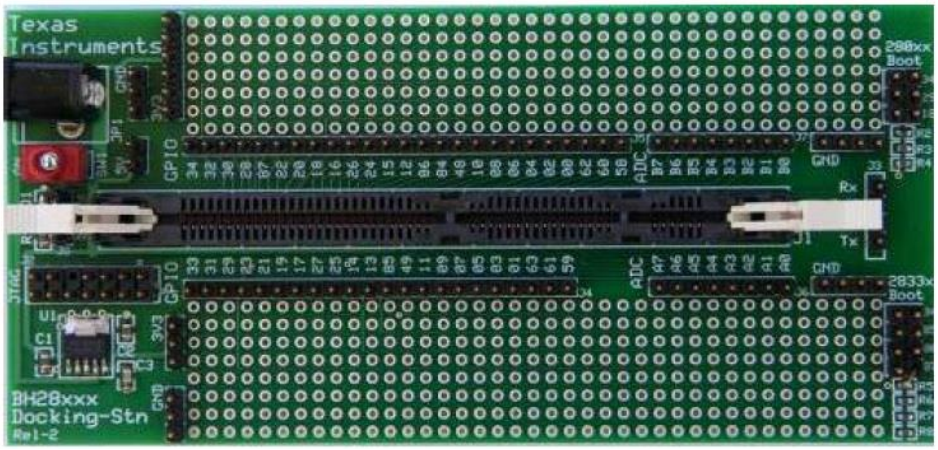


Figura 21 Base de acoplamiento de la familia Delfino C2000.



Figura 22 Tarjeta de Control de TMS320F28335.

4.5.1. Periféricos

Los periféricos integrados del DSP son los siguientes:

6 canales de acceso directo a memoria (DMA).

- Tres Timers de 32 bits.
- Seis módulos PWM mejorados (ePWM1, ePWM2, ePWM3, ePWM4, ePWM5, ePWM6).
- Seis módulos de captura mejoradas (eCAP1, eCAP2, eCAP3, eCAP4, eCAP5, eCAP6).
- Dos módulos mejorados QEP (eQEP1, eQEP2).
- Convertidor analógico digital (ADC) con 16 canales de 12 bits de resolución.
- Dos módulos de controller area network (eCAN), (ECAN) (ECAN-A, ECAN-B).
- tres módulos de interfaz de comunicaciones serial (SCI-A, SCI-B, SCI-C).
- Una interfaz de periféricos (SPI) del módulo serial (SPI-A).
- Un módulo Inter-integrated circuit (I2C).
- Dos módulos multichannel buffered serial port (McBSP-A, McBSP-B).
- Ochenta y ocho puertos de propósito general GPIO (Entrada/Salida).
- Ocho pines de Interrupción Externa (XINTF).
- Para el uso específico del proyecto se hace uso del módulo ADC, seis módulos eCAP, un módulo SCI, puertos GPIO y dos timers.

4.5.2. Conversor análogo digital (ADC)

El módulo ADC tiene 16 canales, 2 módulos configurables independientes ADCA y ADCB cada uno con 8 canales. Los módulos pueden conectarse en forma de cascada para formar un módulo de 16 canales. Aunque tiene múltiples canales de entrada y dos secuenciadores, solo cuenta con un convertidor. La figura 23 muestra el diagrama de bloques del módulo ADC.

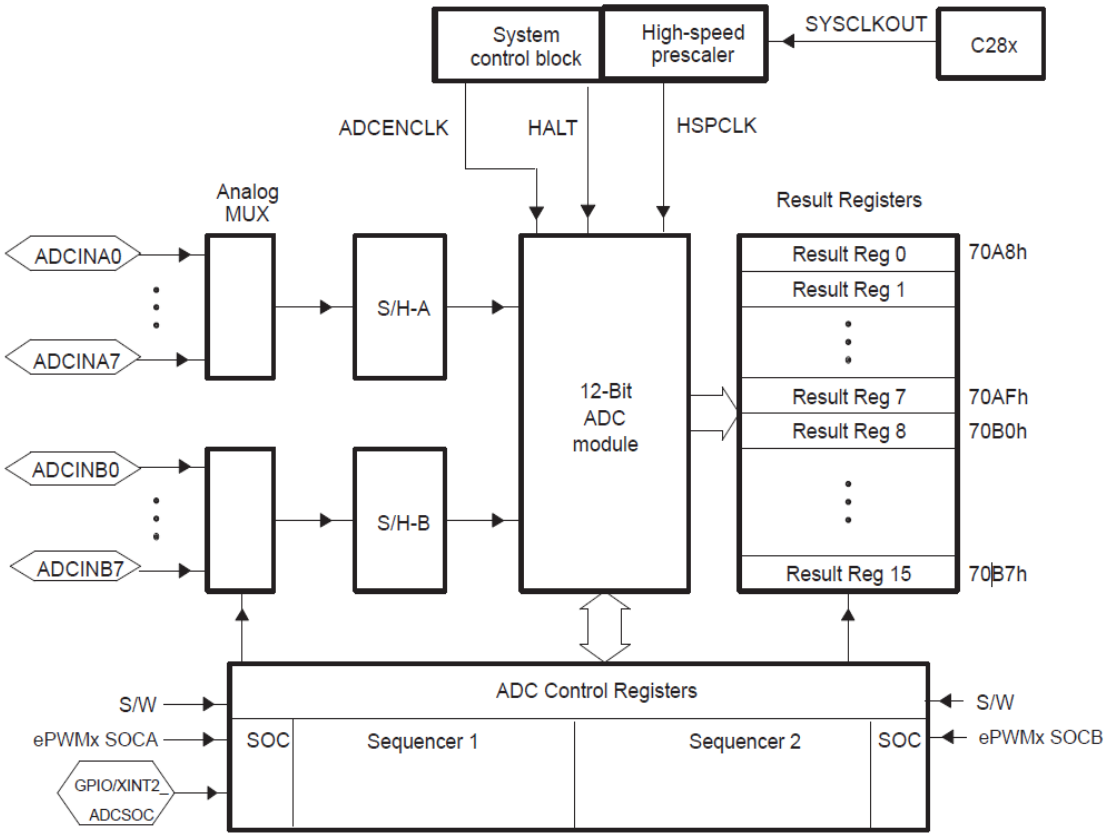


Figura 23 Diagrama de bloques módulo ADC

Funciones del módulo ADC incluyen:

- El núcleo de ADC de 12 bit con una función de doble muestreo y retención (S/H)
- Muestreo simultáneo o secuencial.
- Entrada analógica de 0 a 3v.
- Tiempo de conversión rápida a 12,5MHz, reloj ADC, o 6,25 MSPS.
- 16 canales, entradas multiplexadas con un Mux analógico.
- Proporciona hasta 16 autoconversiones en una sesión, por lo que se puede programar para seleccionar cualquier de los 16 canales.
- El secuenciador puede funcionar como 2 de 8 estados independientes o como uno de 16 estados.
- Tiene 16 registros para almacenar los valores de conversión, ecuación 3.

$$Valor\ digital = 4096\ * \frac{entrada\ analogica - ADCLO}{3}$$

$cuando\ la\ entrada\ es\ de\ 0\ a\ 3v$

Valor digital =0 cuando la entrada es menor a cero.

Valor digital= 4095 cuando la entrada es mayor o igual a 3.

Ecuación 3 Conversión del módulo ADC.

ADCLO = Es un valor de referencia de tensión analógica con el que se puede comprar la señal de entrada.

- Múltiples activadores para la secuencia de inicio de conversión (SOC).
  - S/W software de puesta en marcha inmediata

    -ePWM 1-6

    -GPIO XINT2

- El control de interrupción flexible permite la solicitud de interrupción en cada EOS (parada automática en la secuencia) al final de la secuencia o cualquier otro EOS.
- El secuenciador puede operar en modo Start/stop, lo que permite múltiples disparos de tiempo secuenciado para sincronizar conversiones.
- Activadores de ePWM pueden funcionar de forma independiente en modo de doble secuenciador.

4.5.3. Módulo de captura mejorada (eCAP)

Este módulo cuenta con 6 canales, los cuales pueden trabajar de dos modos: para captura de eventos externos por flancos de subida y/o de bajada, y como generador de señales PWM, en la figura 24 se puede apreciar el diagrama de módulo eCAP.

Los usos de eCAP incluyen:

- Mediciones de la velocidad de la maquinaria de rotación (por ejemplo, piñones dentados detectados mediante sensores Hall)
- Medidas de tiempo transcurrido entre impulsos del sensor de posición.
- Mediciones período y ciclo de trabajo de las señales de tren de impulsos.

- Decodificación de la amplitud de corriente o de tensión derivado del ciclo de trabajo codificado sensores de corriente / tensión.

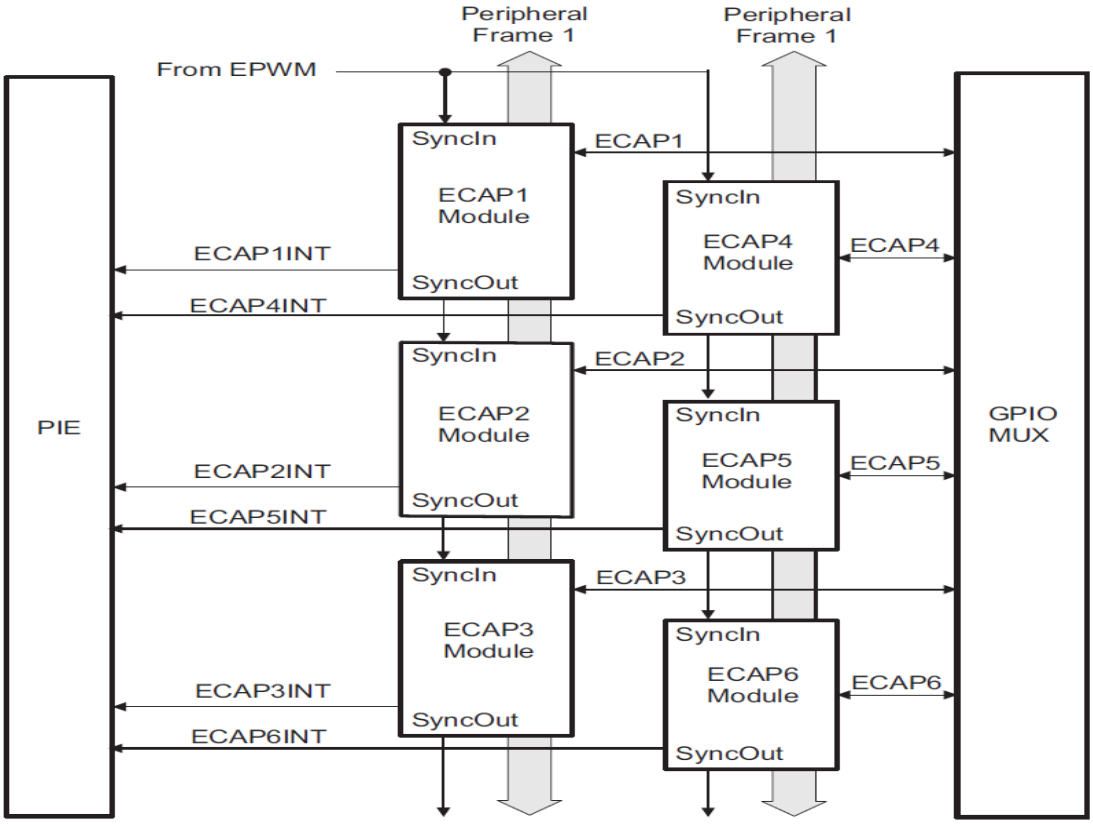


Figura 24 Diagrama de bloques del sistema de múltiples módulos eCAP.

- El módulo eCAP incluye las siguientes características:
- Base de tiempo de 32 bits con una resolución de tiempo 6,67ns con un reloj de 150MHz.
- Registros de captura de marca de tiempo de 4 x 32 bits (CAP 1-CAP 4).
- Secuenciador de 4 etapas (Modulo 4 contador) que se sincroniza a los acontecimientos externos.
- Selección de 4 eventos (falco descendentes/ flancos ascendentes).
- Preescala señal de entrada de captura (de 2-62).
- One-shot comparar registro (2 bits) para congelar las capturas después de 1 a 4 eventos de marca de tiempo.

- El control de las capturas de marca de tiempo continúo utilizando un buffer circular (CAP 1-CAP 4).
- Capacidades de interrupción en cualquiera de los 4 eventos de captura.

4.5.4. Interfaz de comunicación serial (SCI).

La interfaz de comunicaciones serie (SCI) es un puerto serie asíncrono, comúnmente conocido como un UART. Los módulos de SCI soportan comunicaciones digitales entre la CPU y otros periféricos asíncronos que utilizan el formato estándar de no retorno a cero (NRZ). El receptor y el transmisor SCI cuenta cada uno con un FIFO de 16 niveles. Ambos pueden funcionar de forma independiente para la comunicación half-duplex, o simultáneamente para la comunicación full-duplex, en la figura 25 se puede apreciar el diagrama de bloques de SCI.

La velocidad de bits es programable a diferentes velocidades a través de un registro de transmisión de selección de 16 bits.

Características del módulo SCI incluyen:

- Dos pines externos:
  - SCITXD: Contacto de salida de transmisión-SCI
  - SCIRXD: Contacto de recepción de SCI-entrada

    Ambos los pines se pueden utilizar como GPIO si no se utiliza para SCI.
- Velocidad de transmisión programable 64K tasas diferentes
- Formato de datos de palabras
  - Un bit de inicio
  - Longitud de datos de palabras programable de uno a ocho bits
  - Opcional, bit de paridad impar
  - Uno o dos bits de parada
- Cuatro banderas de detección de errores: paridad, invadido y el encuadre.

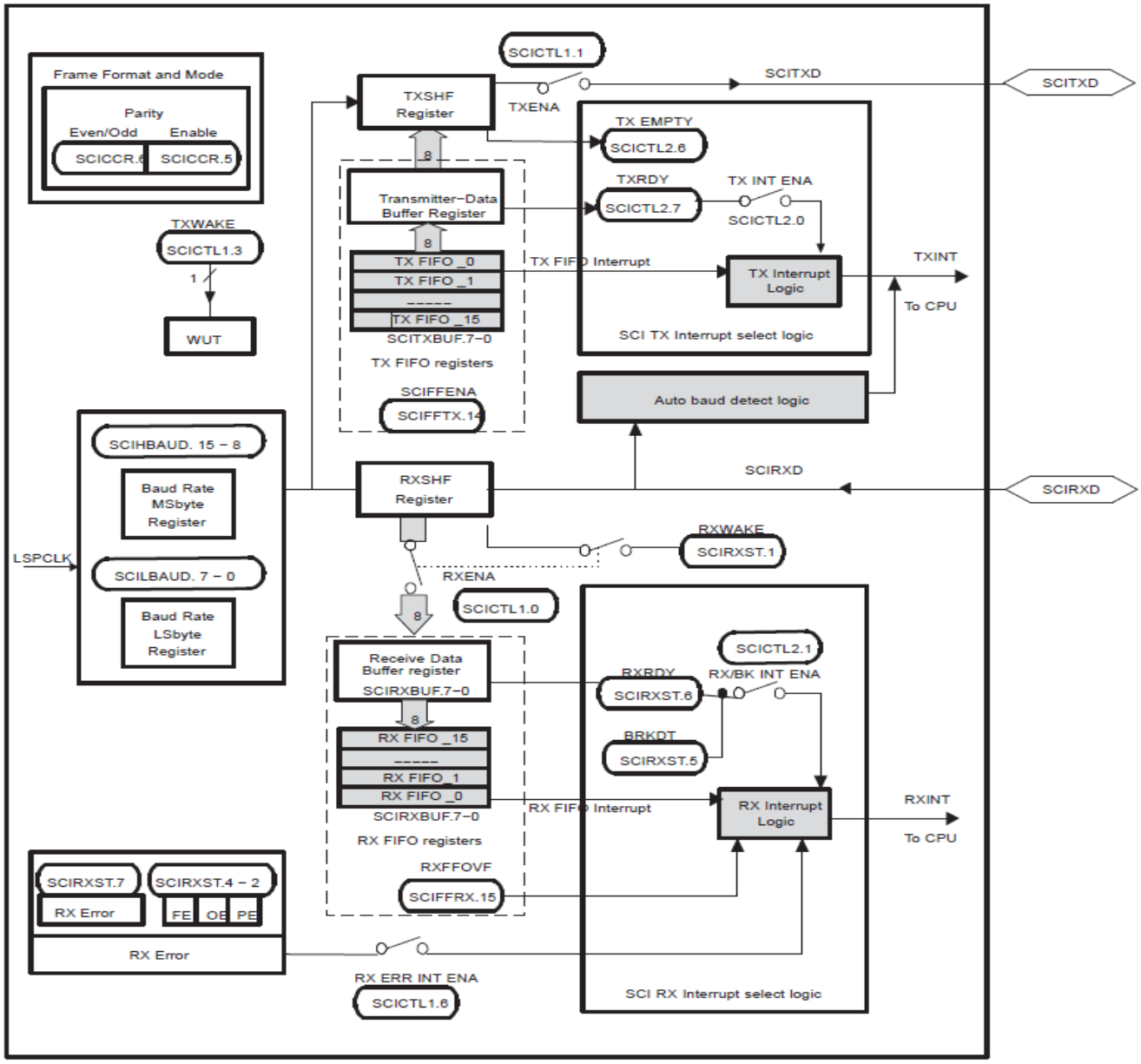


Figura 25 Diagrama de bloques módulo SCI.

4.5.5. Puertos digitales Entrada/Salida (GPIO)

Este módulo cuenta con 88 pines de propósito general los cuales pueden funcionar como entradas o salidas. Su estado alto de 3.3V y su estado bajo de 0V.

Se agrupan en puertos llamados GPIO A, B y C. Cada uno de los pines puede ser utilizado para cuatro funciones distintas, para elegir una función específica se logra empleando un multiplexor.

4.5.6. Timers

El DSP cuenta con tres Timers de CPU de 32 bits. El Timer 2 está reservado para DSP/BIOS. El Timer 0 y el Timer 1 pueden ser usados en aplicaciones, en el caso que no se esté usando el Timer 2 para DSP/BIOS puede usarse para aplicaciones, el diagrama del módulo se puede observar en la figura 26.

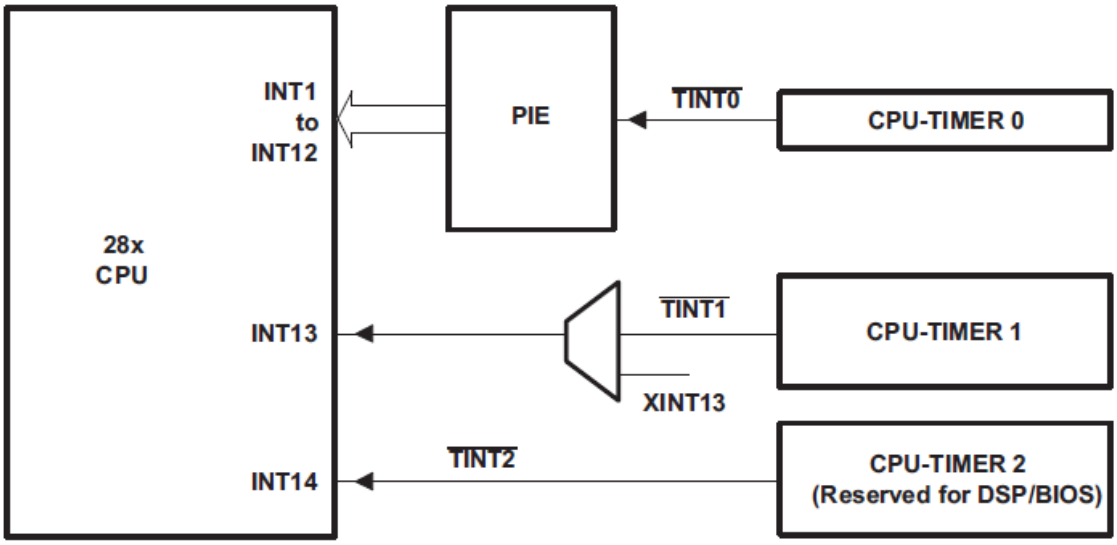


Figura 26 Diagrama de bloques Timers.

4.6. SISTEMA DE COMUNICACIONES

4.6.1 Componentes de la red de comunicaciones

- Puerta de enlace o Coordinador:

Se trata del dispositivo más completo e importante. Su función es almacenar información sobre la red y actuar como su centro de confianza en la distribución de claves de cifrado. Este puede actuar como director de una red en árbol, así como servir de enlace a otras redes. Debe existir uno por cada red.

- Router:

Interconecta los dispositivos separados en la red o que están limitados debido a su cobertura.

- Dispositivo final:

Este dispositivo puede comunicarse con su nodo padre (el coordinador o un router), pero no puede transmitir información destinada a otros dispositivos. De esta forma, este tipo de nodo puede estar dormido la mayor parte del tiempo, aumentando la vida media de sus baterías. [40].

4.6.2 Topologías de red

Es la forma en la que está diseñada la red. La topología de red la determina únicamente la configuración de las conexiones entre nodos. La distancia entre los nodos, las interconexiones físicas, las tasas de transmisión y los tipos de señales no pertenecen a la topología de la red, aunque pueden verse afectados por la misma.

4.6.2.1 Tipos de topologías

Para las redes de sensores se pueden tener tres tipos de topologías como se muestran en la figura 27.

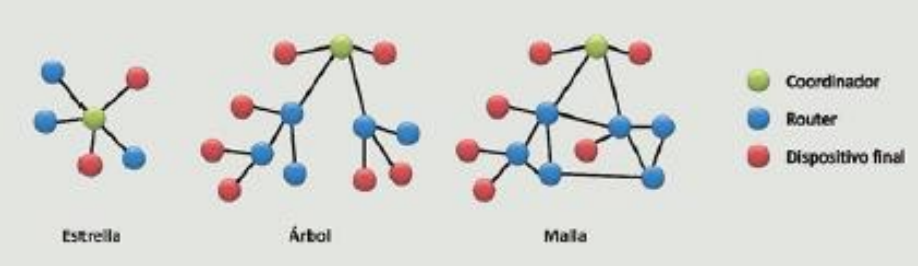


Figura 27 Topologías de red

- Malla: Estas redes están formadas por un coordinador, routers y dispositivos finales. Los nodos pueden establecer comunicación con cualquier otro dispositivo, mediante routers que actúan de repetidores. Se puede considerar que este tipo de topología es una extensión de la topología punto a punto.
- Árbol: Los diferentes componentes de esta red están organizados en una estructura jerárquica. La red está formada por un coordinador y por un grupo de routers de los que pueden colgar, a su vez, otros dispositivos denominados hijos. Éstos pueden actuar como routers con más hijos o como dispositivos finales. Cada nodo sólo puede tener un único padre y los dispositivos finales no pueden tener ningún nodo hijo. Esta topología, al igual que la de Malla, permite expandir la cobertura de la red.
- Estrella: La red está compuesta por varios dispositivos finales conectados únicamente al coordinador central por el que pasan todas las comunicaciones. El alcance máximo de la red viene determinado por el alcance del coordinador, el cual necesita alimentarse a través de la red eléctrica. El resto de dispositivos finales pueden emplear baterías.

4.6.3 Módulos XBEE

Estos módulos son empleados para realizar la comunicación inalámbrica. Son fabricados por diversas empresas como por ejemplo Digi International. Estos dispositivos brindan una variedad de combinaciones de hardware, protocolos, antenas y potencias de transmisión.

Existen varios tipos de módulos XBEE, pero básicamente se pueden agrupar en dos tipos, Serie 1 y Serie 2. Los dos son muy parecidos en el hardware, pero no son compatibles entre

ellos debido al protocolo de comunicaciones que maneja cada uno; los módulos XBEE Serie 1 soportan el protocolo IEEE 802.15.4 y los módulos Serie 2 soportan el protocolo Zigbee, esto se presenta en la tabla 4 de manera más detallada.

	XBEE Series 1	**XBEE Series 2**
Alcance interior (Urbano)	Hasta 30 m	Hasta 40 m
Alcance en exteriores con línea de vista	Hasta 100 m	Hasta 120 m
Potencia de transmisión	1 mW	2mW
Sensibilidad de receptor	-92 dbm (1% PER)	-98 dbm (1% PER)
Corriente de apagado	10 uA	1 uA
Topologías de red	Punto a punto, estrella.	Punto a punto, estrella, malla.
Rango de temperatura	-40 a 85 C	-40 a 85 C
Nº entradas/salidas digitales	8	10
Nº entradas analógicas	7	4
Nº salidas analogicas	2	Ninguna

Tabla 4 comparativa XBee Serie 1 vs Serie 2

4.6.3.1.Capas de protocolos IEEE 802.15.4

Se basa en el modelo OSI, los niveles inferiores los define el estándar como se muestra en la figura 28.

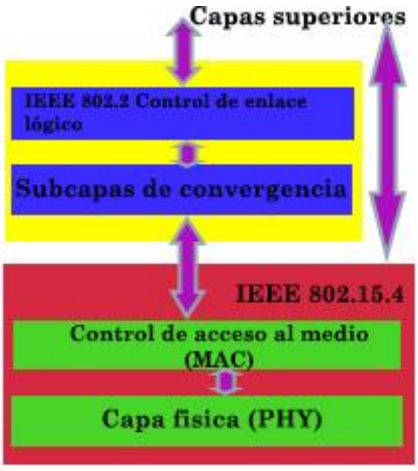


Figura 28 Capas del protocolo IEEE 802.15.4

La capa física (PHY) se encarga del servicio de transmisión de datos sobre el medio físico y la interfaz con la capa de acceso al medio. Opera en una de tres posibles bandas de frecuencia de uso no regulado.

- 868-868,8 MHz: Europa, permite un canal de comunicación (versión de 2003), extendido a tres en la revisión de 2006.
- 902-928 MHz: Norte América, hasta diez canales (2003) extendidos a treinta (2006).
- 2400-2483,5 MHz: uso en todo el mundo, hasta dieciséis canales (2003, 2006).

La capa de control de acceso al medio MAC se encarga de transmitir las tramas MAC usando para ello el canal físico. Ofrece un interfaz de control y regula el acceso al canal físico, controla la validación de tramas, las asociaciones entre nodos, y garantiza slots de tiempo.

4.6.3.2. Capas protocolo Zigbee

De acuerdo a los estándares desarrollados y apropiados por el modelo OSI, el estándar Zigbee crea un modelo de capas presentado en la figura 29. Las primeras dos capas, la física (PHY) y la de acceso al medio (MAC), son definidas por el estándar IEEE 802.15.4. Las capas superiores son definidas por la Alianza Zigbee y corresponden a las capas de red (NWK) y de aplicación (APL) que contienen los perfiles del uso, ajustes de la seguridad y la transmisión de los mensajes.

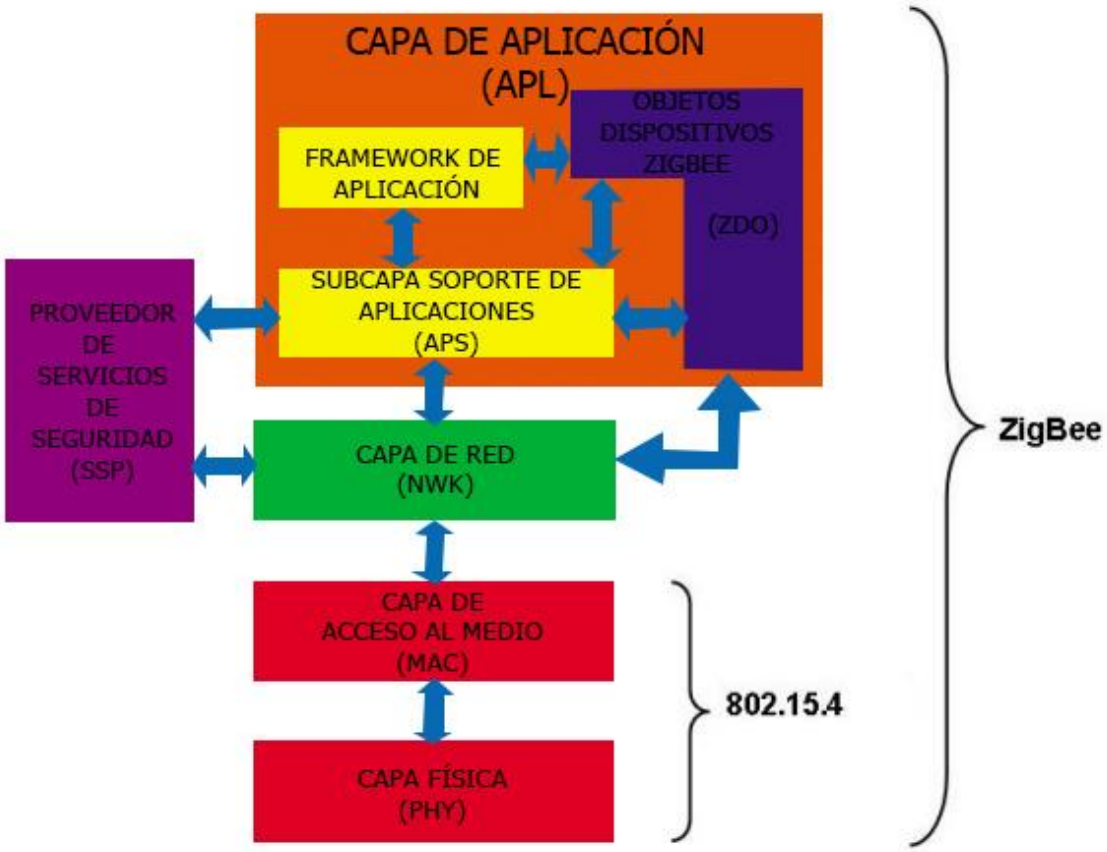


Figura 29 Capas del protocolo IEEE 802.15.4 y Zigbee [52]

La capa de red se encarga de la gestión entre la capa inmediatamente inferior MAC y la superior APL, utilizando dos servicios esenciales para las capas el de control y el de datos, la comunicación entre la APL y la sub-capa MAC, se lleva a cabo en el SAP (Service Access Point) de la capa de Red.

La capa de aplicación se encarga de la definición de los objetos y el uso de las aplicaciones diseñadas para cada uno de los dispositivos integrantes de la red, además brinda el soporte del acceso al usuario y la interfaz de descubrimiento y aceptación de nuevos dispositivos cuando son conectados o incluidos en el sistema.

4.7. HERRAMIENTAS DE SOFTWARE.

4.7.1. Code Composer Studio V6.1.3

Code Composer Studio (CCS) es un herramienta de software propietaria de Texas Instruments (TI) es un entorno de desarrollo integrado (IDE) que permite diseñare implementar aplicaciones para microprocesadores de Texas Instruments, este incluye un conjunto de herramientas utilizadas para desarrollar y depurar aplicaciones embebidas. Incluye un compilador C/C++, editor de código fuente, entorno de creación de proyector, depurador, generador de perfiles y otras características.

La versión 6.1.3 soporta procesadores integrados de las familias, CC13xx, CC2538, CC2650, CC3200, C2000, MSP430, MSP432, TM4C y Hercules [53].

4.7.2. ControSuite TM para microcontroladores C2000 TM

Es un conjunto integrado de herramientas de infraestructura de software, diseñado para minimizar el tiempo de desarrollo de software, este se integra a CCS. Proporciona bibliotecas y ejemplos para cada etapa de desarrollo, lo que es una gran ventaja en el momento que se está aprendiendo a utilizar el microprocesador, ya que de sus ejemplos se pueden basar los proyectos a desarrollar [54].

4.7.3. NetBeans IDE 8.1

La plataforma de netbeans 8.1 es un entorno de desarrollo libre, Que permite el desarrollo de distintas aplicaciones de programación, NetBeans IDE 8.1 proporciona analizadores de código y editores para trabajar con la última versión de Java 8, tecnologías Java SE 8, Embedded Java SE 8, y Java ME Embedded 8. El IDE también tiene apoyo a Maven y Java EE; herramientas para HTML5, en particular para AngularJS; y con PHP y C / C ++ [55].

4.7.4. PosgreSQL 9.5

PostgreSQL es una herramienta de código abierto, que entrega confiabilidad, integridad de datos y la corrección de gran calidad, funciona en distintos sistemas operáticos, incluyendo Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), y Windows. Soporta el uso para claves foráneas, combinaciones, vistas, triggers, procedimientos almacenados y diferentes tipos de datos, incluyendo entero, numérico, Boolean, char, varchar, date, intervalo y timestamp de igual manera permite el almacenamiento de objetos

binarios grandes, como imágenes, sonidos, o de vídeo. Tiene interfaces de programación nativo de C / C ++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre otros.

PostgreSQL cuenta con sofisticadas características tales como control multi-versión de concurrencia (MVCC), punto en el tiempo de recuperación, los espacios de tabla, la replicación asincrónica, transacciones anidadas, respaldos en caliente, un sofisticado planificador de consulta / optimizador, y escribir anticipada registro para tolerancia a fallos. Es compatible con los juegos de caracteres internacionales, la codificación de caracteres de varios bytes, Unicode, y es consciente de la configuración regional de la clasificación, mayúscula y minúscula, y el formato. Es altamente escalable tanto en la enorme cantidad de datos que puede manejar y en el número de usuarios concurrentes que puede acomodar. Hay sistemas activos de PostgreSQL en entornos de producción que manejan más de 4 terabytes de datos [56].

4.7.5. XCTU

Es una aplicación multi-plataforma gratuita, la cual permite a los desarrolladores manejar los módulos de radio frecuencia (RF) de Digi a través de una interfaz gráfica sencilla de usar. En esta aplicación se puede configurar y hacer pruebas de los módulos RF de Digi [57].

4.8. SISTEMA DE IDENTIFICACIÓN.

Identificar la información entre todas las señales transmitidas y recibidas en una red de sensores se convierte en un problema común en este tipo de redes, puesto que el flujo de información llega a ser muy alto, ya sea por la cantidad de variables que se transmiten o por la cantidad de nodos sensores que pertenecen a la red o por la interferencia del medio que se pueda presentar.

Actualmente no existe una norma específica que defina qué sistema de identificación debe usarse para los nodos sensores, se debe hacer uso de las propias técnicas que ofrecen las distintas tecnologías de comunicación.

Particularmente los nodos sensores empleados para las microrredes son fijos, dada las condiciones de operación y funcionamiento de las cargas y las fuentes, esto permite suprimir el uso de dispositivos de localización como GPS, sin embargo, se puede concatenar al número de identificación del nodo sus coordenadas para su localización e incluso las características de la red eléctrica que pertenecen a ese nodo.

5

MODELAMIENTO Y SIMULACIÓN DE LA RED DE SENSORES INALÁMBRICOS

Los acondicionamientos que se describen a continuación manejan un esquema general de funcionamiento, esto permite que la metodología empleada para todos los acoples pueda reproducirse con distintos sensores que compartan el principio de funcionamiento. Así se puede modular el desarrollo del proyecto para que no se vea limitado por los dispositivos de medición elegidos, sino por el contrario que se pueda escalar con facilidad su funcionamiento a cualquier aplicación.

La función de los acondicionamientos es adecuar la señal que entregan los sensores a la tarjeta de procesamiento, las características de la tarjeta se pueden observar en la sección anterior.

5.1. ACOPLE DE TENSIÓN.

El valor que entrega el dispositivo de sesando de tensión es una señal idéntica a la de la red, con una relación de conversión sujeta a las características del mismo, las etapas necesarias para el diseño de este acople son las siguientes:

1. Ajuste de señal: la salida del sensor de tensión es una corriente en función de la tensión de sensado, para un manejo práctico se realiza la conversión a la tensión de la señal, para realizar esto se utiliza la información proporcionada por el fabricante, donde con una sola resistencia se puede realizar esta etapa.
2. Filtro: El funcionamiento interno del sensor asegura un buen aislamiento de la red entregando una señal muy limpia, de igual manera se recomienda el uso de un filtro que elimine cualquier ruido que se pueda llegar a presentar. Como se mencionó en el breve análisis de la red trifásica, hay un desplazamiento entre la señal de tensión y corriente de 30°, en esta etapa se puede corregir dicho desplazamiento con la implementación de un filtro pasa bajos, para el cálculo del valor de tensión de la red, esto no genera ningún inconveniente y por otra parte permite un cálculo más fácil de la distancia de desplazamiento entre la señal de tensión y corriente.
3. Señal de referencia: La señal generada por el sensor es AC y la tarjeta de procesamiento solo permite valores DC, por lo cual es necesario agregar un valor constante DC a la señal sensada, este valor es igual a la mitad del valor máximo permitido por los pines de la tarjeta de procesamiento.
4. Sumador: la señal entregada por la etapa de filtrado y la etapa de referencia se suman para obtener una señal que cumpla con los parámetros de la tarjeta de procesamiento.
5. Acople de impedancias: Esta etapa evita perturbaciones con el pin de la tarjeta al momento de realizar su conexión y limitar su corriente de entrada
6. Protección: Esta etapa está orientada a proteger el pin de conexión frente a algún problema que se pueda generarse, funcionando como un limitador de tensión.

El diagrama de conexión del acople de tensión se puede ver en la figura 30.

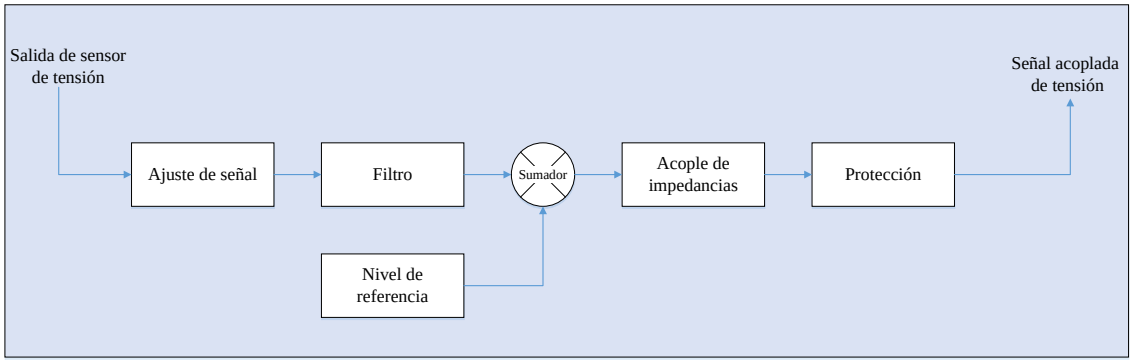


Figura 30 Acople sensor de tensión.

El esquema eléctrico del acople tensión, se replica para cada una de los sensores dispuestos en el esquema de medición, en la figura 31 se presenta el esquema eléctrico para uno de los sensores dispuestos en la red eléctrica. Donde *VsenRS* es la señal que proviene del sensor, *VacpRS* es la señal acoplada, *Vref+* es el nivel de referencia y *FreFRS* es una señal que se emplea para el acople de frecuencia que se presenta más adelante.

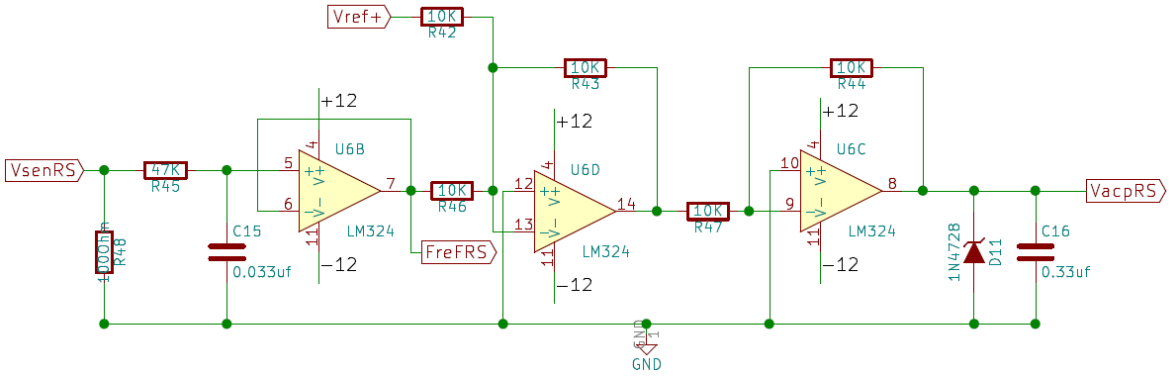


Figura 31 Esquema eléctrico del acople de tensión.

El nivel de referencia del acople de tensión se presenta en la figura 32, este circuito puede utilizarse para el acople de varios sensores de tensión puesto que trabaja de manera independiente a la señal del sensor de tensión aplicando un nivel DC.

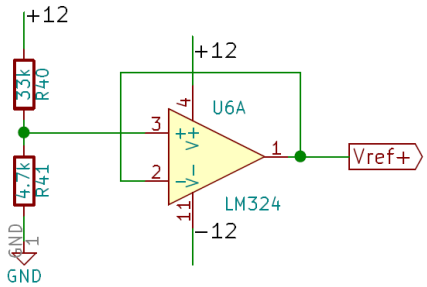


Figura 32 Nivel de referencia.

5.2. ACOPLE DE FRECUENCIA.

El valor de frecuencia de la red se puede extraer con ayuda del sensor de tensión, la señal de entrada de este acople es la que entrega la segunda etapa del acople del sensado de tensión. Las etapas necesarias para el diseño de este acople son las siguientes:

1. Generación PWM: la entrada que proporciona la etapa de filtro del acople del sensado es una señal alterna, con esta señal se genera un PWM que contiene la misma frecuencia de la señal de entrada, siendo el ciclo positivo el tiempo en alto y el ciclo negativo el tiempo en bajo.
2. Ajuste de señal: Se modifica la señal de PWM al rango de trabajo del módulo de medición de la tarjeta de procesamiento y se asegura de igual manera de proteger la tarjeta de procesamiento frente a sobretensiones.
3. Acople de impedancias: Es necesario para evitar que la señal de PWM se deforme al momento de conexión.
4. Protección: Limitador de tención para prevenir daños en la tarjeta de procesamiento.

El diagrama de conexión del acople de frecuencia se puede ver en la figura 33.

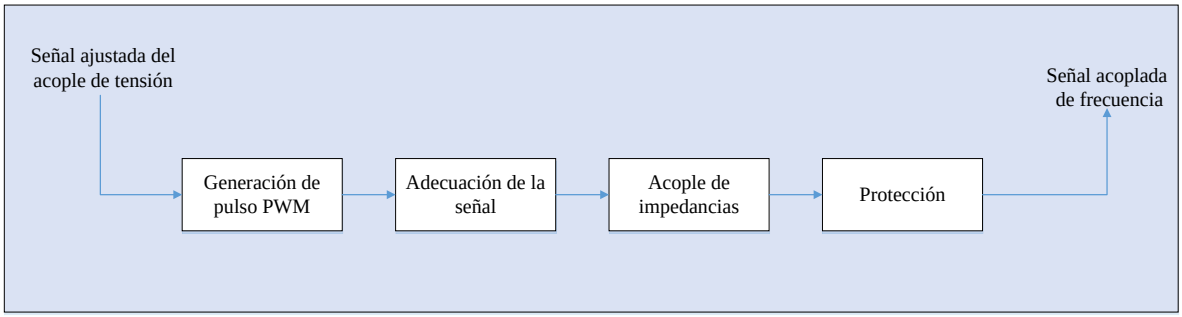


Figura 33 Acondicionamiento de señal de tensión para determinar la variable de frecuencia.

El esquema eléctrico del acople de frecuencia se presenta en la figura 34, donde *FrecVRS* es la señal de salida del acople y *FreFRS* es la señal de que proviene de la segunda etapa del acople del sensor de tensión que será la entrada.

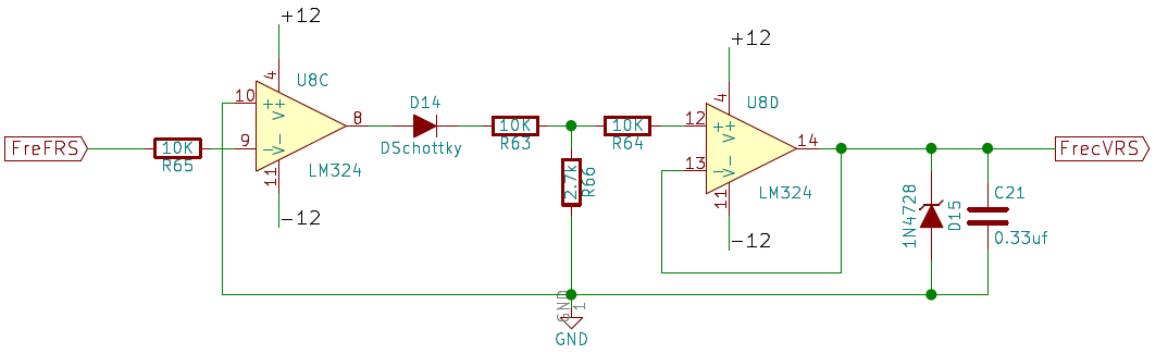


Figura 34 Esquema eléctrico del acople de frecuencia.

5.3. ACOPLE DE CORRIENTE.

El valor que entrega el dispositivo de sensado de corriente es una señal idéntica a la de la red, pero con un nivel de referencia ya incluido, las etapas necesarias para el diseño de este acople son las siguientes:

1. Ajuste de señal: La salida del sensor de corriente es una tensión en función de la corriente de sensado, así que es necesario ajustar esta señal a la ventana de trabajo de la tarjeta de procesamiento y cambiar el nivel de referencia a justo la mitad del límite máximo del pin de lectura.
2. Filtro: De igual manera que el sensor de tensión, la señal que entrega el sensor de corriente es muy limpia, al usar el filtro pasa bajo se asegura eliminar cualquier ruido que se pueda llegar a presentar.
3. Acople de impedancias: Su función es igual a la descrita en el acople de tensión, evita que cuando se realice la conexión la tarjeta se generen perturbaciones en la señal limitando de igual forma la corriente de entrada.
4. Protección: Limitador de tensión para prevenir daños en la tarjeta de procesamiento.

El diagrama de conexión del acople de corriente se puede ver en la figura 35.

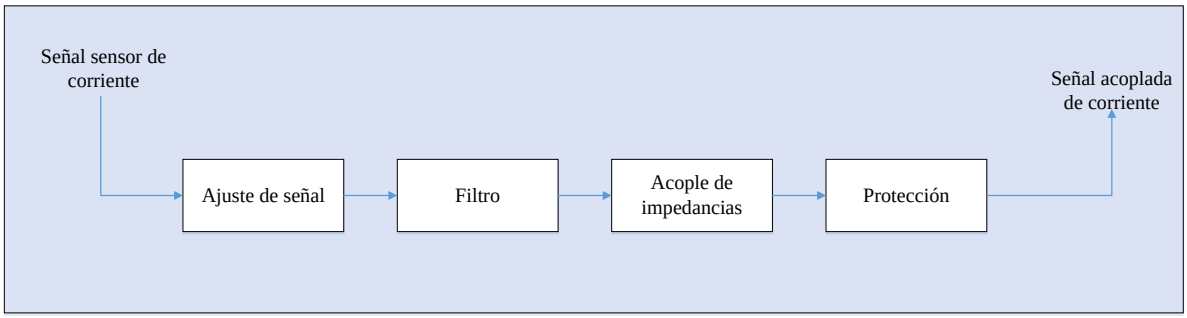


Figura 35 Acople sensor de corriente.

El esquema eléctrico del acople de corriente se presenta en la figura 36, donde *IsenR* es la señal de salida del sensor de corriente y *IacpR* es la señal de acoplada.

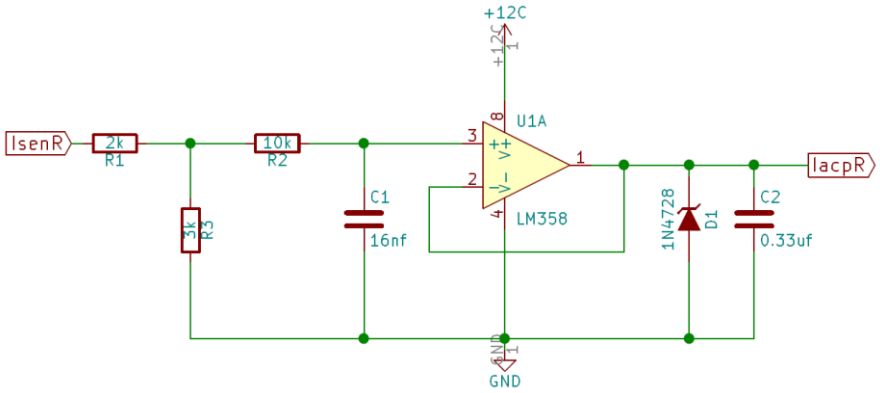


Figura 36 Esquema eléctrico del acople de corriente

5.4. ACOPLE DEL ANGULO DE DESFASE.

La diferencia del ángulo entre las señales de tensión y corriente, será llamado ángulo de desfase. Este valor permite conocer el factor de potencia de la red eléctrica, se hace necesario crear un PWM de la señal de corriente, de esta manera al comparar las señales generadas se obtiene un pulso con la duración en alto que representa el ángulo de desfase. Para este acople se dividirá el esquema en dos partes que se denominaran: acople de frecuencia para la señal de corriente y acople ángulo de desfase.

5.4.1. Acople de frecuencia para la señal de corriente.

La señal generada con este acople tiene como entrada la salida del acople del sensor de corriente, esta es una señal variante con un nivel de referencia. Las etapas necesarias para el diseño de este acople son las siguientes:

1. Eliminación nivel de referencia: La forma de generar un PWM más sencilla, es observando los puntos de cruce por cero de la señal de entrada. La señal de entrada varía alrededor de un nivel de referencia, este valor DC se puede eliminar mediante la suma de este valor, pero de signo contrario o mediante un filtrado que elimine el nivel DC y entregue la señal alterna. El método más fiable es mediante el filtrado ya que al utilizar un restador de señal el nivel de referencia no es exacto debido a la tolerancia de los componentes, deformando la señal que se quiere generar y afectar de manera considerable la medición.
2. Ganancia: Es necesario agregar una ganancia a la señal alterna generada en la etapa anterior para poder observar de manera más clara el cruce por cero.
3. Generación PWM: la señal alterna se compara con la tierra del sistema, obteniendo como resultado una señal de PWM donde su ciclo útil en alto corresponde a la duración del ciclo positivo de la señal de corriente de la red.
4. Ajuste de señal: Se modifica la señal de PWM al rango de trabajo de los componentes de la segunda parte del esquema.
5. Acople de impedancias: Se asegura que la señal de PWM no se deforme al comprar la señal con la segunda parte del esquema.

El diagrama de conexión del acople de frecuencia para la señal de corriente se puede ver en la figura 37.

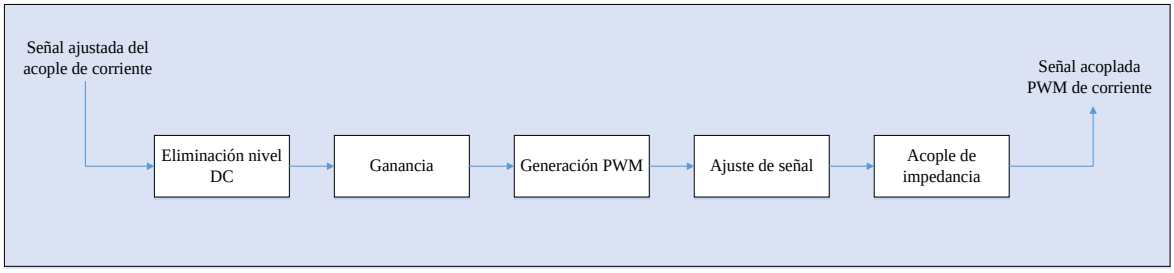


Figura 37 Acondicionamiento de señal de corriente para genera PWM.

El esquema eléctrico del acondicionamiento de la señal de corriente para la generación del PWM se presenta en la figura 38. Donde *IacpR* es la señal de entrada, que es la misma señal entregada por el acople del sensor de corriente y *FreR* es la salida, que será la señal PWM generada.

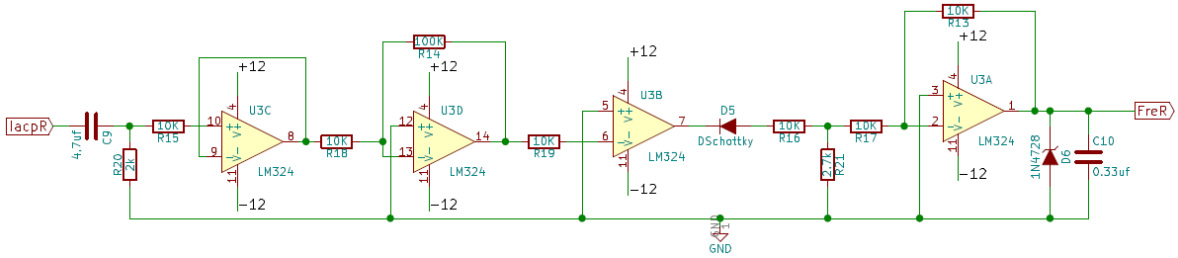


Figura 38 Esquema eléctrico del acondicionamiento de la señal de corriente para generar el PWM.

5.4.2. Acople ángulo de desfase.

Se genera una señal de PWM con un ciclo útil que varía según el ángulo de desfase que se tenga en las señales de corriente y tensión. Las etapas necesarias para el diseño de este acople son las siguientes:

1. Comparación de señales PWM: Como se evidencia en el capítulo del análisis de la red eléctrica, el ángulo de desfase es la distancia del cruce por cero de la señal de tensión hasta el cruce por cero de la señal de corriente, partiendo de que los PWM´s su ciclo en alto pertenece al semiciclo positivo y el ciclo en bajo a el semiciclo negativo, si se invierte la señal de PWM de tensión y se compara con la señal de PWM de corriente, el tiempo en que las dos señales de PWM están en alto coincide con el desfase. De esta manera se tiene una señal de PWM con ciclo útil variante que está en función del ángulo de desfase.
2. Ajuste de señal: Se modifica la señal de PWM variante en ciclo útil, al rango de trabajo del módulo de medición de ciclo útil de la tarjeta de procesamiento.
3. Protección: Limitador, para la protección por sobretensión al pin de lectura dela tarjeta de procesamiento.

El diagrama de conexión del acople ángulo de desfase se puede ver en la figura 39

En la figura 40 se presenta el diagrama eléctrico que entrega la señal para determinar el desfase de las señales. *FrecVRS* y *FreR* son las señales de entradas, *FrecVRS* proviene de la señal de generada en el acople de frecuencia y *FreR* es la señal de PWM generada a partir de la señal de corriente que se muestra en la figura 34. *FpR* es la señal de salida del esquema y es la que se dirige a la tarjeta de procesamiento.

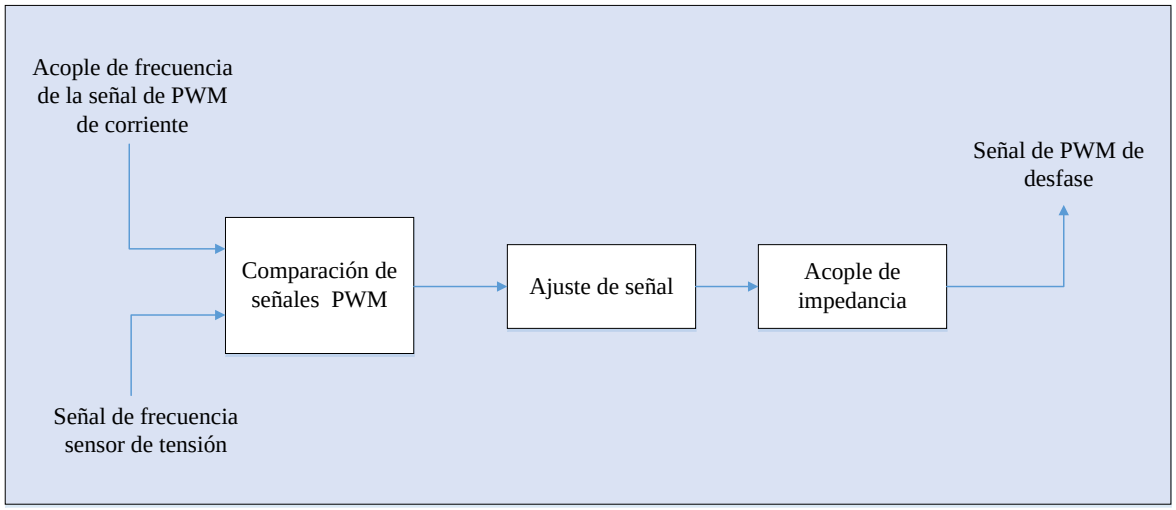


Figura 39 Obtención y acondicionamiento de señal PWM del desfase entre la tensión y corriente.

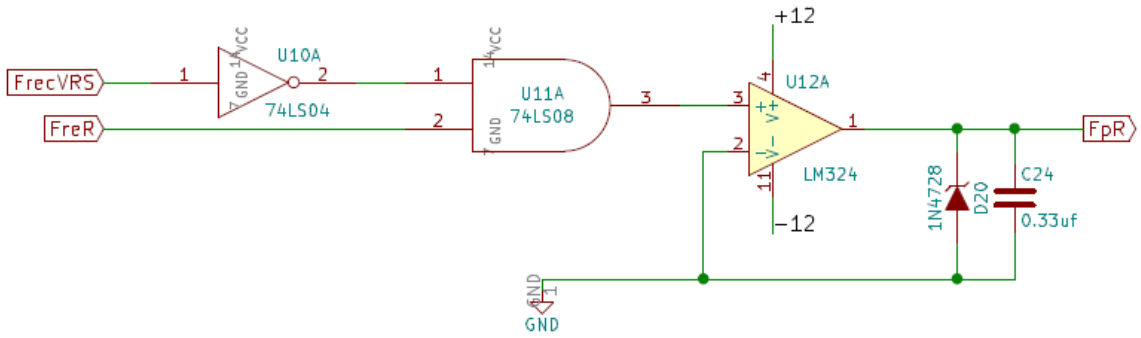


Figura 40 Señal de desfase entre la tensión y la corriente.

5.5. ESQUEMA COMPLETO DE ACONDICIONAMIENTO DE SEÑALES.

Los acoples mencionados anteriormente son necesario para cada uno de los sensores, teniendo en cuenta la topología de medición a desarrollar es necesario implementar estos acoples para cada uno de los sensores ubicados en la red trifásica, en el caso de la señal de la línea de neutro solo es necesario realizar el acople respectivo a la señal de corriente y no es necesario realizar ningún acople de desfase. De igual manera el diagrama completo es necesario para cada nodo de sensado de la microrred.

El esquema completo de sensado por nodo se puede ver la figura 41. El esquema eléctrico completo de acople se puede ver en el anexo 11.2.

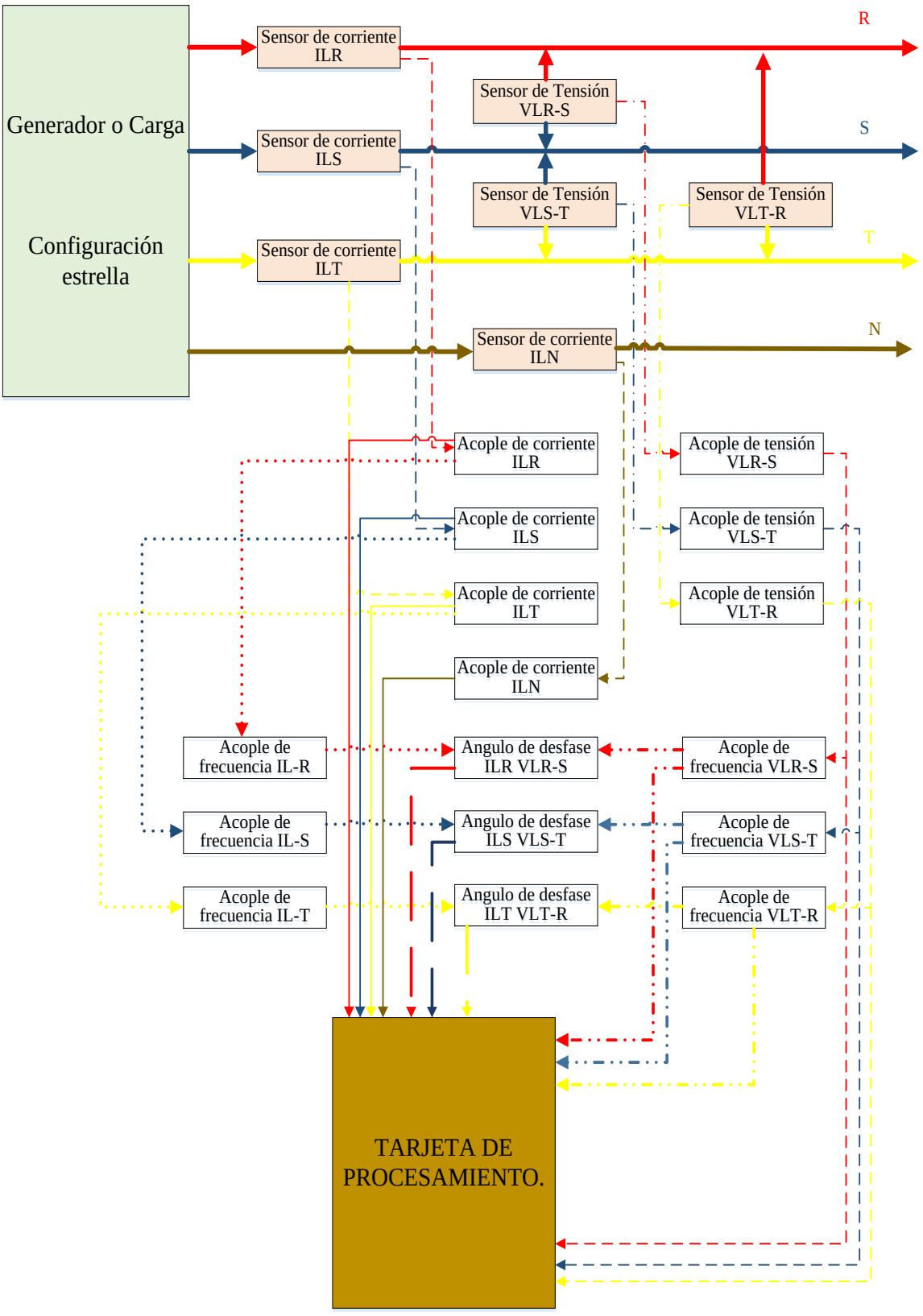


Figura 41 Esquema general de acondicionamiento de señales para el nodo sensor.

5.6. PROCESAMIENTO Y CALCULO DE VARIABLES.

En esta sección se describe el diseño del software de la tarjeta de adquisición. Para la adquisición y cálculos de las variables de medición, tensión RMS de línea, corrientes RMS de línea, frecuencia de las tensiones de línea, fases de tensiones de línea y ángulo de desfase entre tensiones de línea y corrientes de línea, fue necesario realizar pruebas preliminares de manera independiente, en esta sección se describe como fueron diseñadas.

5.6.1. Medición de tensiones y corrientes.

Dado a los requerimientos del proyecto, se necesita que la adquisición de las señales de tensión línea a línea y corrientes de línea se den en el mismo instante, se configura el módulo ADC para que trabaje en modo de muestreo simultáneo y el secuenciador trabaje en cascada en 7 de sus 16 canales; 3 para tensiones de línea a línea (VLR-S, VLS-T y VLT-R); 4 para corrientes de línea (ILR, ILS, ILT y ILN).

Haciendo pruebas se determinó que por espacio de memoria es necesario configurar la velocidad de muestreo para que en un vector de 185 posiciones se guarden los valores muestreados de un periodo de una señal de máximo 60 Hz, con estos valores se logra hallar las tensiones y corrientes RMS.

5.6.1.1. Configuración de velocidad de muestreo

1. Para que el registro prescaler del reloj de alta velocidad (HISPCP) trabaje a 25MHz. Se debe tener en cuenta que se relacionan el valor del bit 0 (HSPCLK) del registro HISPCP con la velocidad de núcleo del DSP (SYSCLKOUT) de la siguiente forma, ecuación 4.

$$Sí\ HISPCP \neq 0, \quad HSPCLK = SYSCLKOUT/(HISPCP\ x\ 2).$$

$$Sí\ HISPCP = 0, HSPCLK = SYSCLKOUT.$$

Ecuación 4 Relación del registro del reloj

Para que la velocidad sea de 25MHz el valor del registro HISPCP es de 3.

```
#define ADC_MODCLK 0x3 // HSPCLK = SYSCLKOUT/2*ADC_MODCLK2 = 150MHz/(3*2)   =
                                  25.0 MHz
   SysCtrlRegs.HISPCP.all = ADC_MODCLK;    // HSPCLK = SYSCLKOUT/ADC_MODCLK
```

2. Se configura la velocidad del reloj central del ADC (ADCLKPS) para que sea de 1,38MHz. Con esta velocidad se almacena un periodo de la señal, en un vector de 185 posiciones. La relación de reloj del ADC es HSPCLK / (ADCCLKPS x 2).

Por lo que en este caso el valor del registro ADCCLKPS es de 9; para que la división del prescaler sea de 18, por lo tanto la velocidad de muestreo sería de 1,38MHz.

```
#define ADC_CKPS   0x9   // ADC module clock (ADCLKPS) = HSPCLK/2*ADC_CKPS   =
                            25.0MHz/(9*2) = 1.38MHz
                     AdcRegs.ADCTRL3.bit.ADCCLKPS = ADC_CKPS;
```

3. Se configura el periodo de la ventana de muestras para que sea de 11,59ms.

Para lo cual la relación de los bits [8:12] (ACQ_PS) del registro ADCTRL1 y el ADCLKPS, ecuación 5.

$$ACQ_PS + 1 \,/\, ADCLKPS$$

Ecuación 5 Configuración del periodo.

Por lo que el valor del registro ACQ_PS es de 15.

```
#define ADC_SHCLK  0xf   // S/H VENTANA DEL PERIODO DEL MODULO ADC          =
                            16 ADC clocks
                   AdcRegs.ADCTRL1.bit.ACQ_PS = ADC_SHCLK;
```

5.6.1.2. Configuración modo de muestreo y secuenciador

1. Para configurar que el modo de muestreo sea simultáneo el bit 0 (SMODE_SEL) del registro ADCTRL3 debe estar en 1.

```
AdcRegs.ADCTRL3.bit.SMODE_SEL = 1;
```

2. Para que el secuenciador funcione en cascada el bit 4 (SEQ_CASC) del registro ADCTRL1 debe estar en 1.

```
AdcRegs.ADCTRL1.bit.SEQ_CASC = 1
```

3. Se asocia el secuenciador a usar para la conversión con el canal de entrada analógica. Por ejemplo AdcRegs.ADCCHSELSEQ1.bit.CONV00 = 0x0, configura la conversión de ADCINA0 y ADCINB0, y los resultados de estos canales se almacenan en los registros de la siguiente forma; ADCINA0 en el registro ADCRESULT0 y ADCINB0 en el registro ADCRESULT1.

```
AdcRegs.ADCCHSELSEQ1.bit.CONV00 = 0x0;   // ADCINA0 (ADCRESULT0) y
                        ADCINB0 (ADCRESULT1)
AdcRegs.ADCCHSELSEQ1.bit.CONV01 = 0x1;   // ADCINA1 (ADCRESULT2) y
                        ADCINB1 (ADCRESULT3)

AdcRegs.ADCCHSELSEQ1.bit.CONV02 = 0x2;   // ADCINA2 (ADCRESULT4) y
                        ADCINB2 (ADCRESULT5)
AdcRegs.ADCCHSELSEQ1.bit.CONV03 = 0x3;   // ADCINA3 (ADCRESULT5) y
                        ADCINB3 (ADCRESULT5)
```

5.6.1.3. Medición y calibración, para tensiones y corrientes

El ADC de 12 bits, traduce la salida de los acoples de tensión y corriente que es de 0 a 3V a un valor digital de 0 a 4095. Ya que la señal de entrada es una señal alterna con un nivel DC, se debe hallar el valor máximo traducido de la entrada para asociarlo al valor de entrada de la fuente.

En el diagrama de flujo de la figura 42, se muestra el diseño de cómo realizar la medición para la calibración, y así poder asociar de manera más precisa en los valores de entrada con los de salida.

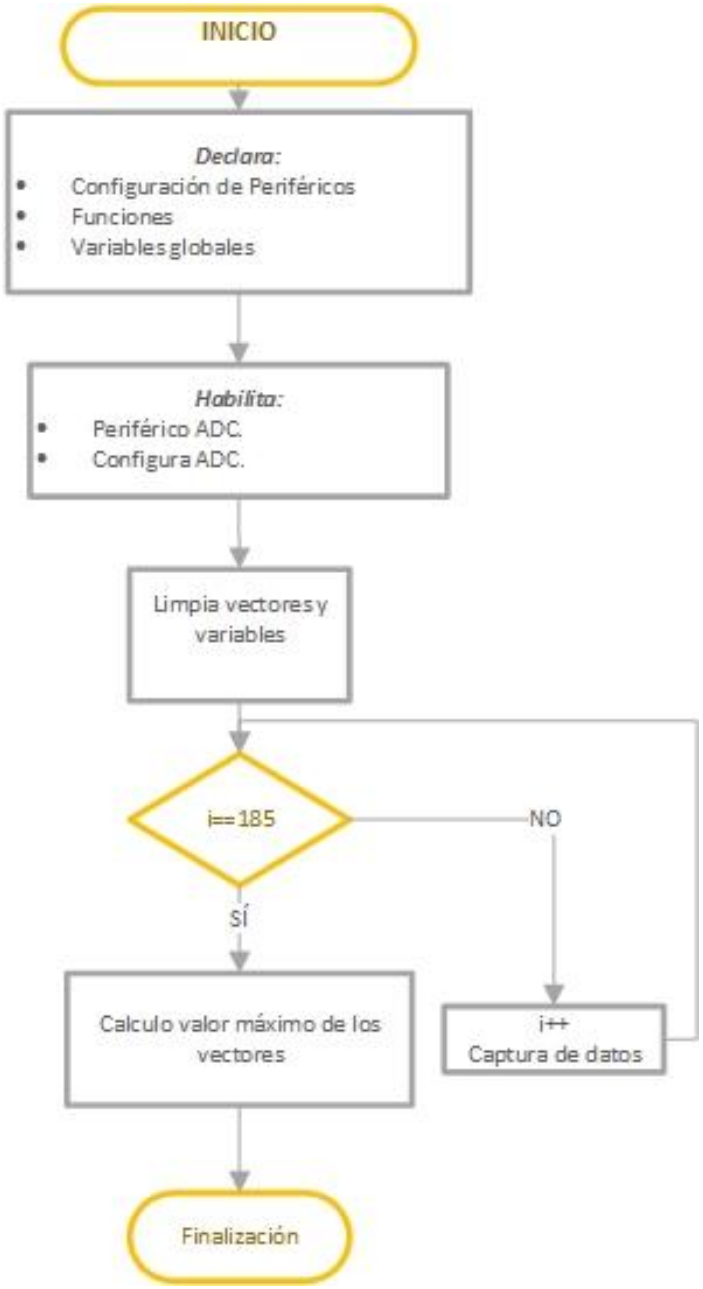


Figura 42 Diagrama de flujo prueba para calibración

5.6.1.1.1. Calibración tensión

Se parte de que el comportamiento de los sensores es lineal por lo que se toman dos valores de tensión de entrada de la fuente (X1 y X2 puntos de la recta), siendo X2 mayor a X1. Se halla la pendiente que genera los dos valores máximos traducidos y se multiplica por un factor (*Fmul*) que es de 0.0007326, este valor representa el factor de conversión del valor adquirido en el módulo ADC a su equivalente en voltios de 0 a 3v, ecuación 6.

$$m = \frac{valor\ maximo\ traducido\ de\ X2 * Fmul + - valor\ maximo\ traducido\ de\ X1 * Fmul}{X2 - X1}$$

Ecuación 6 pendiente entre dos tensiones traducidos

Conociendo está pendiente se puede relacionar el valor de conversión necesario con la siguiente formula, ecuación 7.

$$voltaje\ conv = \sqrt[2]{2} * (\left(\frac{(voltaje\ \ de\ entrada\ traducido * Fmul) - valor\ maximo\ traducido\ de\ X2 * Fmul}{m}\right) + X1)$$

Ecuación 7 tensión de conversión

5.6.1.1.2. Calibración corriente

Al igual que la calibración de la tensión, en la corriente se toman 2 valores de corriente X1 y X2, se hallan sus valores máximos y por medio de las siguientes ecuaciones se pueden relacionar los valores de entrada.

Se halla la pendiente que genera los dos valores máximos traducidos multiplicador por el factor multiplicativo (Fmul), ecuación 8.

$$m = \frac{valor\ maximo\ traducido\ de\ X2 * Fmul + - valor\ maximo\ traducido\ de\ X1 * Fmul}{X2 - X1}$$

Ecuación 8 pendiente entre dos corrientes traducida

Conociendo está pendiente se puede relacionar el valor de conversión necesario con la fórmula que se presenta en la ecuación 9.

$$corriente\ conv = \sqrt[2]{2} * (\left(\frac{(corriente\ \ de\ entrada\ traducido * Fmul) - valor\ maximo\ traducido\ de\ X2 * Fmul}{m}\right) + X1)$$

Ecuación 9 corriente de conversión

5.6.2. Calculo de valores RMS

- **Cálculo de tensiones RMS**

Conociendo el valor de conversión de la tensión de entrada, para hallar los valores RMS se hace uso de las ecuaciones 10, 11 y 12.

$$Voltaje\ DC = \sum_{i=1}^{184} \frac{voltaje\ conv[i]}{184}$$

Ecuación 10 Tensión DC.

$$Voltaje\ rms\ con\ DC = \sqrt{\sum_{i=1}^{184} \frac{(voltaje\ conv[i])^2}{184}}$$

Ecuación 11 Tensión RMS con nivel DC

$$Voltaje\ RMS\ true = \sqrt{((Voltaje\ rms\ con\ DC)^2 - (Voltaje\ \ DC)^2)}$$

Ecuación 12 Tensión RMS

- **Cálculo corrientes RMS**

Conociendo el valor de conversión de la corriente de entrada, para hallar los valores RMS se hace uso de las ecuaciones 13, 14 y 15.

$$Corriente\ DC = \sum_{i=1}^{185} \frac{corriente\ conv[i]}{184}$$

Ecuación 13 Corriente DC

$$Corriente\ RMS\ con\ DC = \sqrt{\sum_{i=1}^{185} \frac{(corriente\ conv[i])^2}{184}}$$

Ecuación 14 Corriente RMS con DC.

$$Corriente\ RMS\ true = \sqrt{((Corriente\ rms\ con\ DC)^2 - (Corriente\ \ DC)^2)}$$

Ecuación 15 Corriente RMS

En el diagrama de flujo presentado en la figura 43, se muestra el diseño para poder realizar el cálculo de los valores RMS de tensiones de línea y corrientes de línea.

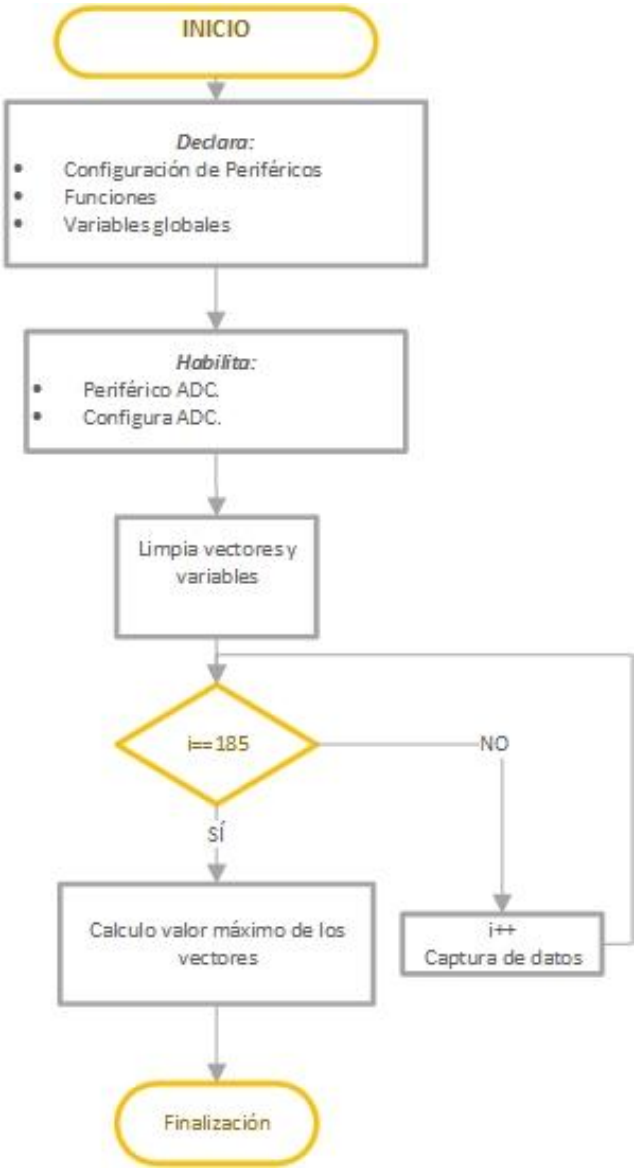


Figura 43 Diagrama de flujo prueba cálculo de valores RMS

5.6.3. Medición de frecuencias, fases y Angulo de desfase entre tensiones de línea a línea y corrientes de línea.

Dado a que es necesario conocer el valor de frecuencia de las tensiones línea a línea y el ángulo de desfase que se produce entre las tensiones y corrientes, se hace uso de los módulos eCAP, los cuales sirven para capturar eventos externos ya sea por flanco de subida y/o bajada.

Para hallar las frecuencias y fases de las tensiones se usan los módulos eCAP1-eCAP3; y el ángulo de desfase entre corrientes y tensiones se utiliza los módulos eCAP4-eCAP6.

Los módulos eCAP1-eCAP3 se configuran para que la interrupción de cada uno se dé cuando se detecte un flanco de subida y que se capturen 4 de estos eventos. Como se muestra en la

figura 44, una vez se active el contador de cada módulo, se guarda en un registro de un evento CAP1 el valor que tiene el contador una vez se detecte el franco de subida, para el segundo evento se guarda en CAP2, para el tercero en CAP3 y para el cuarto en CAP4.

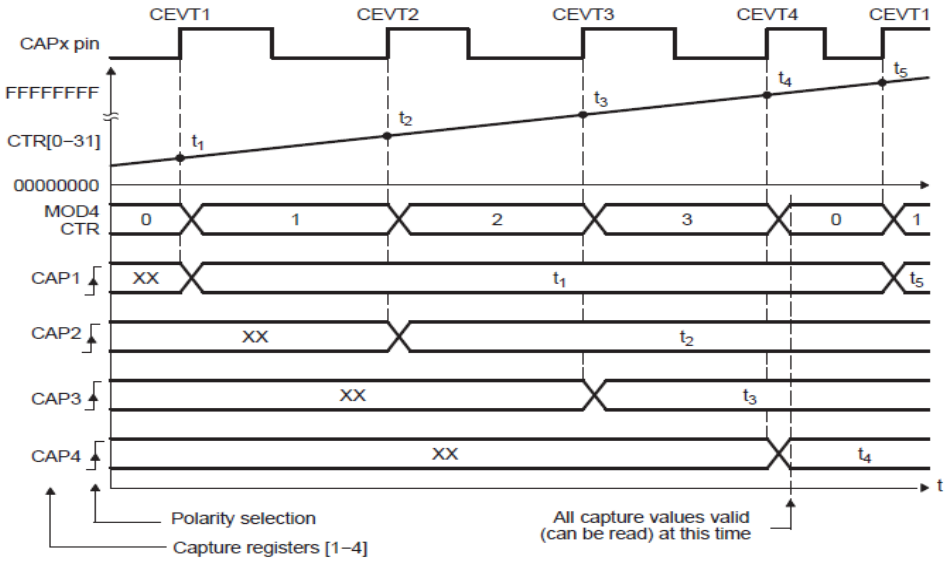


Figura 44 Captura por detección flanco de subida y modo Absolute time-stamp.

Los módulos eCAP4-eCAP6 se configuran para que la interrupción de cada uno se dé cuando se detecte un flanco de subida y de bajada, que se capturen 4 de estos eventos. Como se muestra en la figura 45, una vez se active el contador de cada módulo, se guarda en un registro de un evento CAP1 el valor que tiene el contador una vez se detecte el franco de subida, para el segundo evento flanco de bajada se guarda en CAP2, para el tercero flanco de subida en CAP3 y para el cuarto flanco de bajada en CAP4.

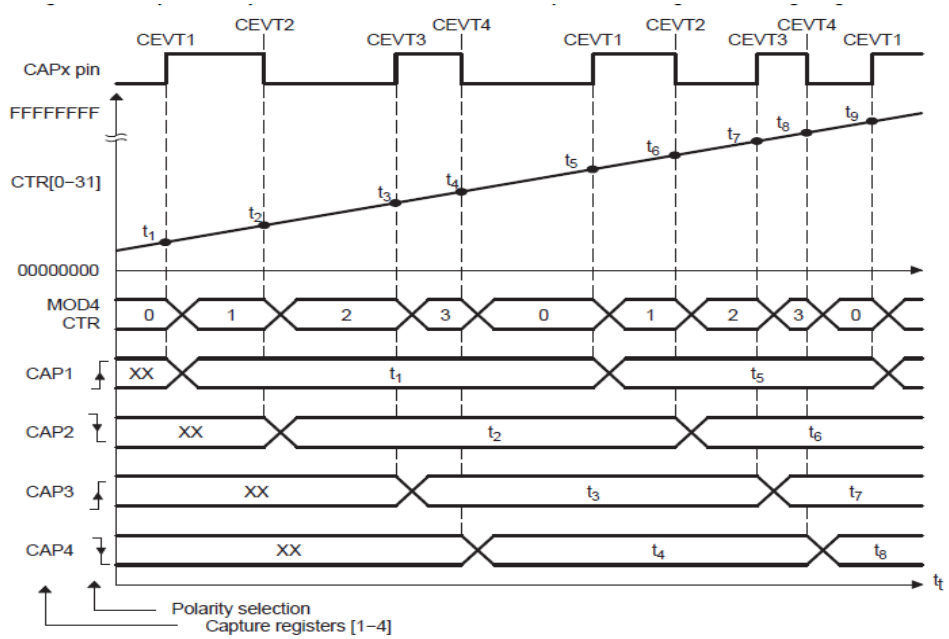


Figura 45 Captura por detección flanco de subida / bajada y modo Absolute time-stamp.

5.6.3.1. Configuración de los módulos eCAP1, eCAP2 y eCAP3, para medir frecuencias y fases.

Para que los módulos funcionen detectando el franco de subida los bits 0, 2, 4 y 6 del registro ECCTL1 de cada uno de los módulos deben estar en 0. Para que funcione en el modo de Absolute Time-Stamp bits 1, 3, 5 y 7 deben estar en 0 y para que estos 4 eventos de captura se den se habilita el bit 8 (CAPLDEN).

```
ECap1Regs.ECCTL1.bit.CAP1POL = 0;          // Rising edge bit 0
ECap1Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge bit 2
ECap1Regs.ECCTL1.bit.CAP3POL = 0;          // Rising edge bit 4
ECap1Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge bit 6
ECap1Regs.ECCTL1.bit.CTRRST1 = 0;          // Absolute time-stamp. bit 1
ECap1Regs.ECCTL1.bit.CTRRST2 = 0;          // Absolute time-stamp. bit 3
ECap1Regs.ECCTL1.bit.CTRRST3 = 0;          // Absolute time-stamp. bit 5
ECap1Regs.ECCTL1.bit.CTRRST4 = 0;          // Absolute time-stamp. bit 7
ECap1Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4
```

5.6.3.2. Configuración de los módulos eCAP4, eCAP5 y eCAP6, para medir ángulos de desfase entre voltajes línea a línea y corrientes de línea.

Para que los módulos funcionen detectando el franco de subida y bajada, los bits 2 y 6 del registro ECCTL1 deben estar en 0; y los bits 1 y 4 del registro ECCTL1 deben estar en 1. Para que funcione en el modo de Absolute Time-Stamp bits 1, 3, 5 y 7 deben estar en 0 y para que estos 4 eventos de captura se den se habilita el bit 8.

```
ECap5Regs.ECCTL1.bit.CAP1POL = 1;          // Falling edge bit 1
ECap5Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge  bit 2
ECap5Regs.ECCTL1.bit.CAP3POL = 1;          // Falling edge bit 4
ECap5Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge  bit 6
ECap5Regs.ECCTL1.bit.CTRRST1 = 0;          // Absolute time-stamp. Bit 1
ECap5Regs.ECCTL1.bit.CTRRST2 = 0;          // Absolute time-stamp. Bit 3
ECap5Regs.ECCTL1.bit.CTRRST3 = 0;          // Absolute time-stamp. Bit 5
ECap5Regs.ECCTL1.bit.CTRRST4 = 0;          // Absolute time-stamp. Bit 7
ECap5Regs.ECCTL1.bit.CAPLDEN = 1;         // Enable CAP1-CAP4
```

5.6.3.3. Calculo de frecuencias, fases y ángulos de desfase entre tensiones línea a línea y corrientes de línea.

- **Frecuencia**

La resta de los valores de los registros CAP2-CAP1 Y CAP4-CAP3, dan como resultado el valor el tiempo que tarda en ocurrir un flanco de subida; ya que la frecuencia del módulo es de 150MHz, la relación para hallar la frecuencia se presenta en la ecuación 16.

$$frecuencia = \frac{150M}{CAP2 - CAP1}$$

Ecuación 16 Calculo de Frecuencia.

- **Fase**

Para hallar la fase de las tensiones se toma como referencia la entrada del módulo eCAP1 Y timer1 que se activa cada 1µs y cuenta cuanto tiempo tarda en detectarse en flanco de subida en los módulos eCAP2 y eCAP3 con referencia al eCAP1.

Este tiempo que tardan tiene una relación con respecto a la fase en la que se encuentran. Y esta relación es la ecuación 17, en señales con una frecuencia de 60 Hz.

$$fase = \left(\frac{contador\ timer1 - 1006}{28.91666}\right) + 120$$

Ecuación 17 Calculo de Fase.

- **Angulo de desfase**

La variación de ciclo de trabajo de las señales de entrada representan un ángulo de desfase entre tensiones y corrientes, por lo que la resta de los valores de los registros CAP3-CAP1 y CAP3-CAP2, dan como resultado el valor el tiempo que tarda en ocurrir un ciclo en alto y un ciclo en bajo respectivamente; ya que la frecuencia del módulo es de 150MHz, la relación para hallar el Angulo de desfase se presenta en la ecuación 18.

$$angulo\ desfase = \left(\frac{(CAP3 - CAP2) * 360}{CAP3 - CAP4}\right)$$

Ecuación 18 Calculo de ángulo de desfase

En el diagrama de flujo que se presenta en la figura 46, se muestra como fue el diseño para realizar la captura y cálculo de las frecuencias de las tensiones de línea, fases de las tensiones y ángulos de desfase entre tensiones de línea y corrientes de línea.

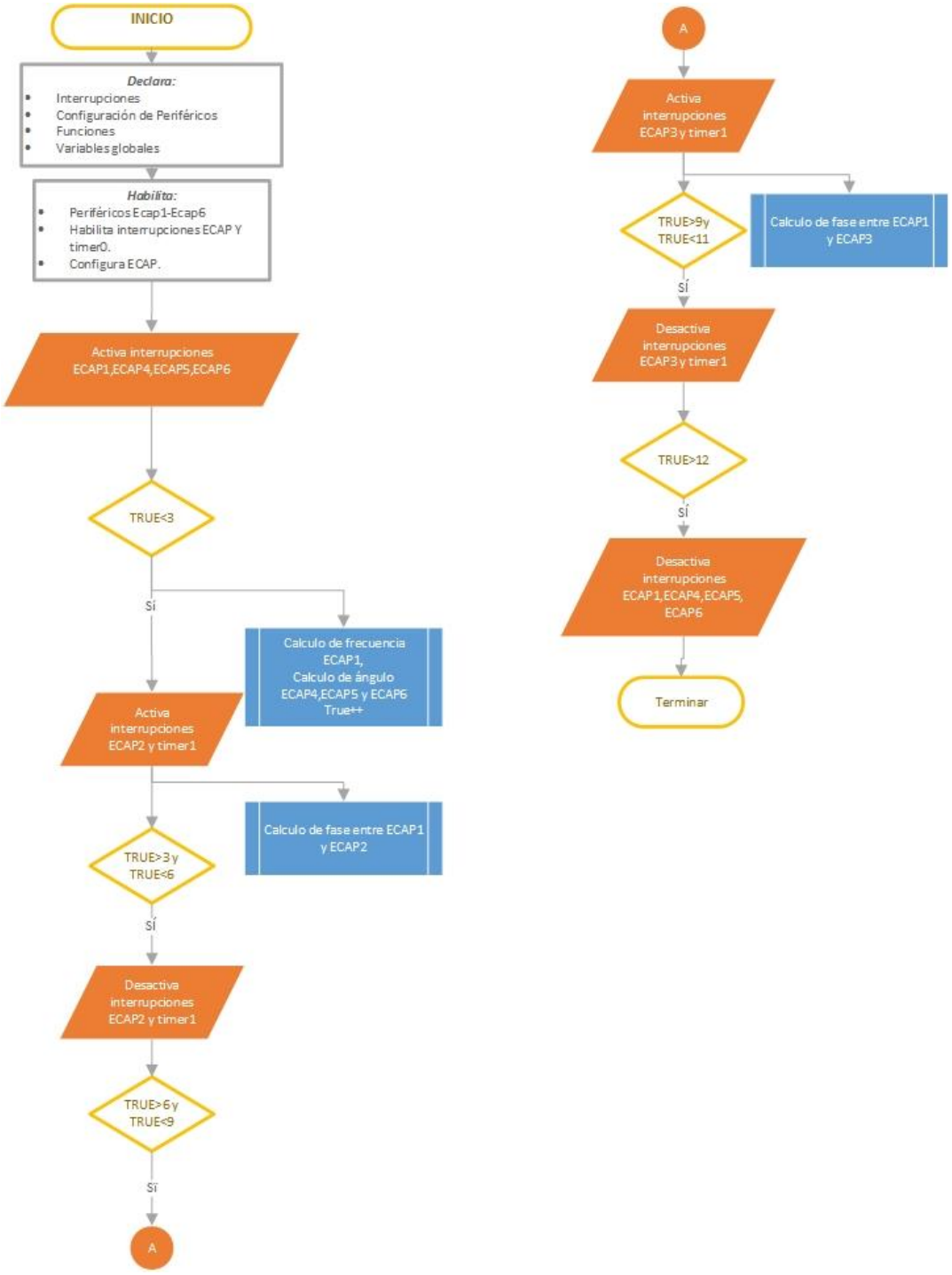


Figura 46 Diagrama de bloques prueba valores de frecuencia, fase y ángulo de desfase.

5.7. ENVIÓ Y RECEPCIÓN DE DATOS.

Para el envío y recepción de datos, se configura el módulo SCIB, para que genere una interrupción cuando al buffer de SCIBRX le llegue algún dato y cuando sea requerido se envíen los datos al nodo concentrador.

La información adquirida a partir del procesamiento de las señales de los sensores, debe ser transmitida al nodo de concentrador, él será el encargado de almacenar y tratar la información.

Se debe asegurar que la información que sea transmitida al nodo concentrador pueda analizarse sin problemas, para ello se realiza una codificación de las variables. Esta trama realizada debe tener un orden estricto ya que es necesario que en el nodo concentrador se realice un proceso de decodificación del paquete enviado que solo puede realizarse conociendo como están distribuidas las variables en la trama

Otro factor importante en él envió de información es la confirmación del paquete recibido, para evitar que la información se pierda y no llegue al destino o llegue de manera incorrecta, es necesario implementar un mecanismo de confirmación. Una vez el nodo concentrador pueda identificar toda la información de manera correcta, tiene la labor de enviar al nodo sensor una señal para que él pueda detener él envió de información, si el nodo sensor no recibe esta confirmación asumirá que la información se ha extraviado, por lo que retransmitirá la trama de información hasta que se le notifique su correcta recepción evitando la pérdida de información.

En el diagrama de la figura 48, se muestra el diseño para la prueba de envío y recepción de datos en el módulo SCIB. El esquema eléctrico de la conexión entre la tarjeta de procesamiento y el dispositivo de comunicación Xbee se presenta en la figura 47, donde *TX* y *RX* son los pines de comunicación, *+3.3* y *GND* los pines de polarización.

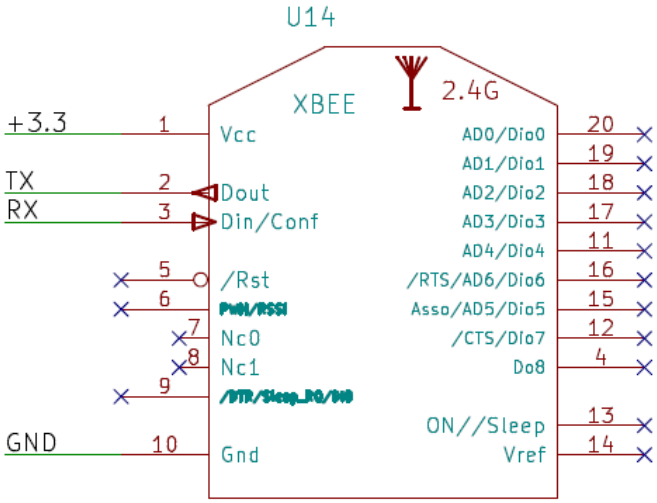


Figura 47 Esquema de conexión entre el dispositivo de comunicación y la tarjeta de procesamiento.

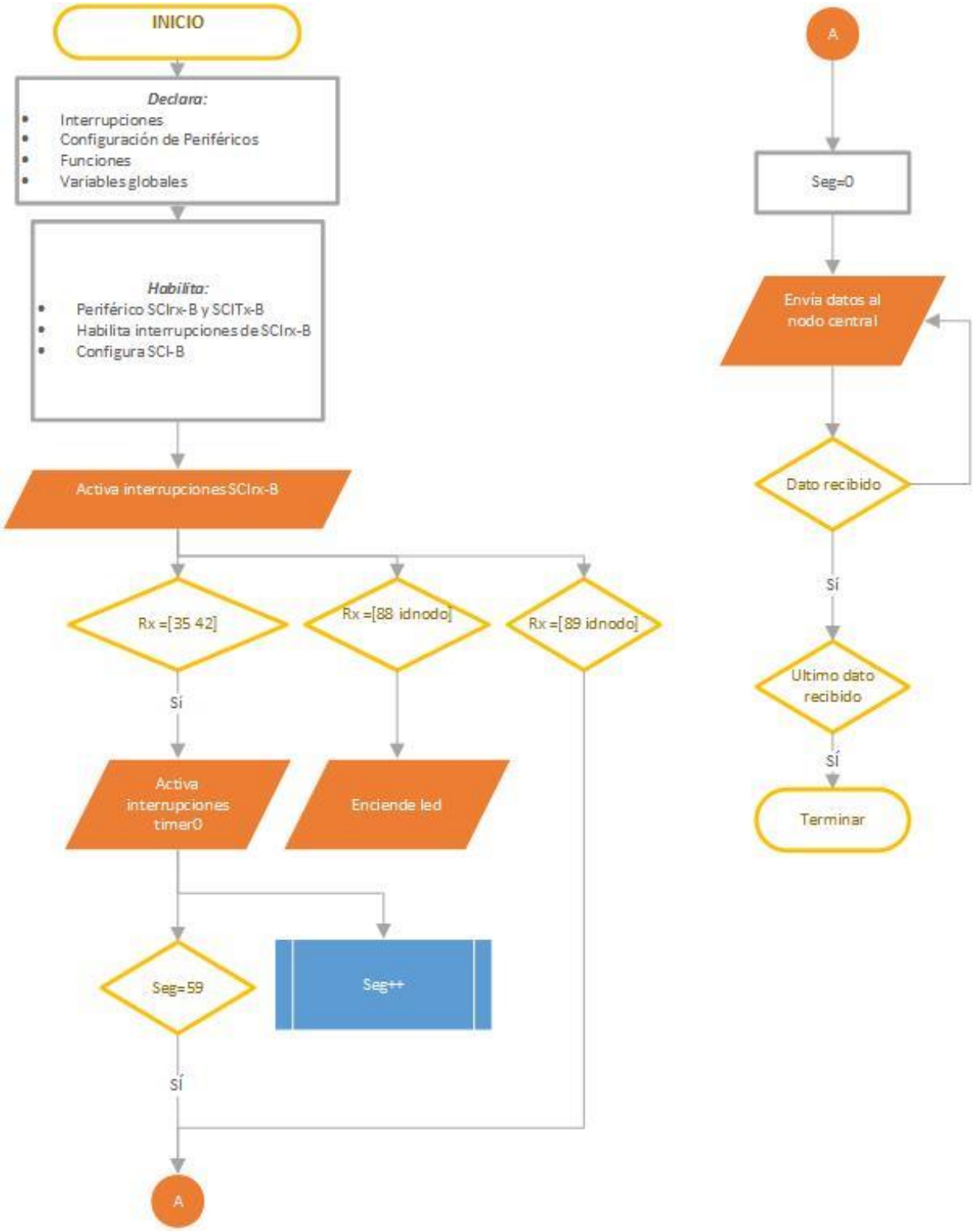


Figura 48 Diagrama de flujo prueba envío y recepción de datos

5.7.1. Diseño del software del nodo sensor

En base al diseño de las pruebas realizadas, se realiza el diseño general del software del nodo sensor.

La información a enviar son tensiones RMS de línea, corrientes RMS de línea, frecuencia de tensiones de línea, fases de tensiones de línea y ángulo de desfase entre tensiones de línea y corrientes de línea. La trama a enviar con la información debe incluir, identificador de nodo, tipo de valor a enviar, día, mes, año, hora y min en el que se adquirió y envío la información.

La trama enviada se ensambla como se muestra en la figura 49.

60	ID Nodo	Tipo	Día	Mes	Año parte1	Año parte2	Hora	Minuto	Valor entero1	Valor entero2	Valor entero3	Decimal 1	Decimal 2	62
14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Figura 49 Trama a enviar

- Los valores 60 y 62 son los indicadores de inicio y final de la trama.
- ID nodo, es el número identificador único de cada nodo, pueden tener valores de 0 a 5.

El capo Tipo, es un número identificador que correspondiente a la clase de variables calculadas, como se indican en la tabla 5.

Tipo	Número de identificación
Tensión RMS VLR-S	1
Tensión RMS VLS-T	2
Tensión RMS VLT-R	3
Corriente RMS ILR	4
Corriente RMS ILS	5
Corriente RMS ILT	6
Corriente RMS ILN	7
Fase2	8
Fase3	9
Frecuencia VLR-S	10
Frecuencia VLS-T	11
Frecuencia VLT-R	12
Angulo desfase VLR-S vs ILR	13
Angulo desfase VLS-T vs ILS	14
Angulo desfase VLT-R vs ILT	15

Tabla 5 Campo Tipo de valor y su correspondiente número identificador

- Campo Día, es el número del día en el que se tomó el dato y puede tener valores de 1-31.

- Campo Mes, es el número del mes en el que se tomó el dato y puede tener valores de 1-12.
- Campo Año, es el número del año en el que se tomó el dato, este está en dos partes. La primera indica los primeros dos dígitos del año y puede tener valores de 0-59. La segunda parte corresponde a los 2 últimos dígitos del año y puede tener valores de 0-59.
- Campo Hora, es el número de la hora en la que se tomó el dato.
- Campo Minuto, es el número del minuto en que se tomó el dato.

El valor correspondiente al valor calculado del tipo de dato. Este se divide en 5 dígitos los primeros 3 correspondientes al valor de la unidad, y los dos últimos el decimal correspondiente. El nodo concentrador recibe los datos en código ASCII y son convertidos a entero, por lo que se debió tener en cuenta que este código solo tiene 255 símbolos, y si algún valor de campo supera el 255, cuando se realice la conversión no tendrá ningún valor en entero. Por ello el campo año se divide en dos partes cada una de a 2 dígitos y el valor calculado en 5 dígitos. Ninguno de los dígitos del valor calculado debe coincidir con los indicadores de inicio y final de trama, por lo que se asegura que al dividir lo en 5 dígitos nunca tome los valores 60 y 62.

En el diagrama de la figura 50, se puede observar el esquema general de procesamiento realizado por la tarjeta ADSP, para la adquisición, cálculos y envío de la información al nodo concentrador.

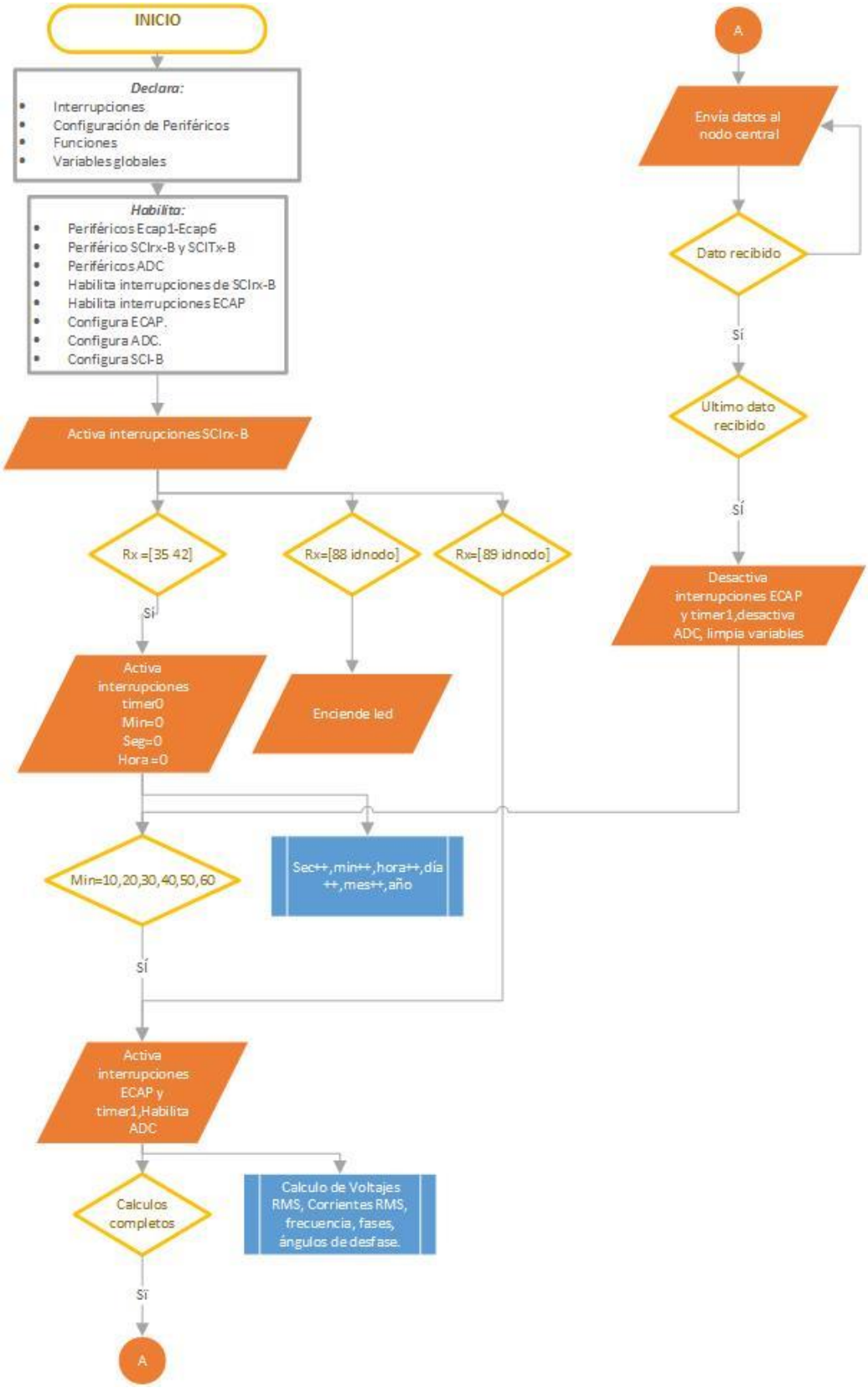


Figura 50 Esquema general de procesamiento y comunicación del nodo sensor



Figura 51 Esquema de conexión de tarjeta de procesamiento.

El esquema de conexión de la tarjeta de procesamiento se presenta en la figura 51 donde:

- *IacpR, IacpS, IacpT* y *IacpN,* son las señales acopladas de los sensores de corriente.
- *VacpRS, VacpST* y *VacpRT,* son las señales acopladas de los sensores de tensión.
- *FrecVRS, FrecVST* y *FrecVRT,* son las señales acopladas de frecuencia.
- *FpS, FpT* y *FpR* son las señales de los acoples de desfase de corriente tensión.
- *Tx* y *Rx*, son las señales de comunicación.
- *LEDA* y *LDEP*, son indicadores
- *RCK, TMS, EMU1, TDI, TDO, TRSTn* y *EMU0,* son los pines de JTAG.
- *+5 y GND*, son las señales de polarización.

5.7.2. Diseño del software del nodo concentrador

El nodo concentrador es el encargado de recibir la información proveniente de la red de nodos sensores, para esto se desarrolla una herramienta que permita entablar un puente de comunicación con los nodos sensores.

Las siguientes son las tareas requeridas para este caso del nodo concentrador.

- Recepción de la información de los nodos sensores.
- Clasificación y almacenamiento de la información en la base de datos.
- Petición de información a los nodos sensores.
- Consulta y visualización de la información almacenada.
- Sincronización de la red de nodos.
- Calculo de potencias P, Q y S.
- Envió de indicador al nodo correspondiente en el caso que la frecuencia se menor a 59.6 Hz y mayor a 60.6 Hz.

5.7.2.1. Calculo de potencias

Una vez los 15 tipos de variables el nodo central las haya recibido y almacenado, se calcula las potencias P, Q y S por línea y se almacena en la base de datos.

Las potencias por línea se calculan por medio de las siguientes ecuaciones 19.

$$\text{Potencia activa (P)línea R} = \text{VRS} * \text{IR} * \cos\theta$$

$$\text{Potencia activa (P)línea S} = \text{VST} * \text{IS} * \cos\theta$$

$$\text{Potencia activa (P)línea T} = \text{VTR} * \text{IT} * \cos\theta$$

$$\text{Potencia aparente (S)línea R} = \text{VRS} * \text{IR} * \text{sen}\,\theta$$

$$\text{Potencia aparente (S)línea S} = \text{VST} * \text{IS} * \text{sen}\,\theta$$

$$\text{Potencia aparente (S)línea T} = \text{VTR} * \text{IT} * \text{sen}\,\theta$$

$$\text{Potencia reactiva } (Q)\text{línea R} = \text{VRS} * \text{IR}$$

$$\text{Potencia reactiva } (Q)\text{línea S} = \text{VST} * \text{IS}$$

$$\text{Potencia reactiva } (Q)\text{línea T} = \text{VTR} * \text{IT}$$

Ecuación 19 Ecuaciones de potencia por línea.

Al igual que los valores enviados por el nodo sensor, se identifican el tipo de valores calculados en el nodo central por un número de identificación como se muestra en la tabla 6. De la misma forma estos valores recibidos son almacenados en la base de datos.

Tipo	Número de identificación
Potencia activa (P) línea R	16
Potencia activa (P) línea S	17
Potencia activa (P) línea T	18
Potencia reactiva (Q) línea R	19
Potencia reactiva (Q) línea S	20
Potencia reactiva (Q) línea T	21
Potencia aparente (S) línea R	22
Potencia aparente(S) línea S	23
Potencia aparente (S) línea T	24

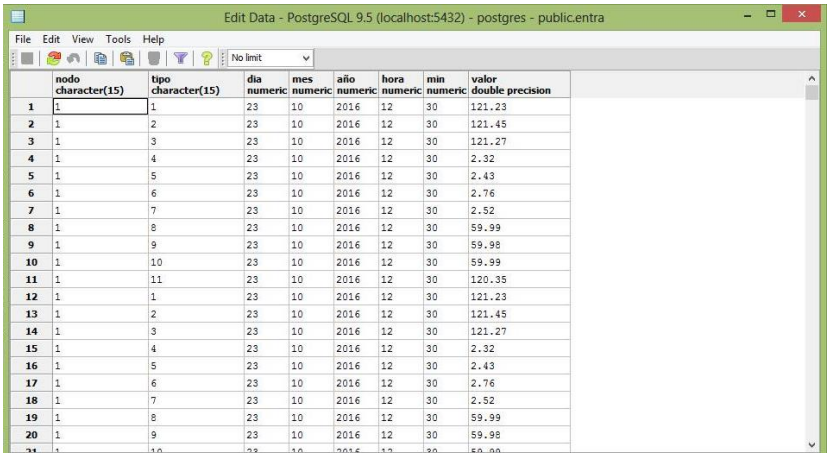
Tabla 6 Identificación de valores calculados

5.7.2.2. Base de datos

Mediante la herramienta PostgreSQL se crea una tabla en la base datos, con las características de variable y tipo que se presentan en la tabla 7, en la figura 52 se puede ver la tabla generada en la base de datos.

Variable	Tipo
nodo	character(15)
tipo	character(15)
día	Numeric
mes	Numeric
año	Numeric
hora	Numeric
min	Numeric
valor	double precision

Tabla 7 Tabla de variables y tipo, para la base de datos



	nodo character(15)	tipo character(15)	dia numeric	mes numeric	año numeric	hora numeric	min numeric	valor double precision
1	1	1	23	10	2016	12	30	121.23
2	1	2	23	10	2016	12	30	121.45
3	1	3	23	10	2016	12	30	121.27
4	1	4	23	10	2016	12	30	2.32
5	1	5	23	10	2016	12	30	2.43
6	1	6	23	10	2016	12	30	2.76
7	1	7	23	10	2016	12	30	2.52
8	1	8	23	10	2016	12	30	59.99
9	1	9	23	10	2016	12	30	59.98
10	1	10	23	10	2016	12	30	59.99
11	1	11	23	10	2016	12	30	120.35
12	1	1	23	10	2016	12	30	121.23
13	1	2	23	10	2016	12	30	121.45
14	1	3	23	10	2016	12	30	121.27
15	1	4	23	10	2016	12	30	2.32
16	1	5	23	10	2016	12	30	2.43
17	1	6	23	10	2016	12	30	2.76
18	1	7	23	10	2016	12	30	2.52
19	1	8	23	10	2016	12	30	59.99
20	1	9	23	10	2016	12	30	59.98

Figura 52 Base de datos generada en el PostgreSQL

5.7.2.3. Desarrollo de la Aplicación.

La aplicación está compuesta de manera general en las siguientes partes.

1. Conexión

La información proveniente del sistema de comunicación se conecta de manera directa al programa, este se encarga de organizar la información proveniente de los nodos sensores y de establecer la conexión con la base de datos generada, con la conexión establecida por el programa se envía a la base de datos la información recibida.

De forma paralela al almacenamiento de información también se realiza la labor de cálculo de potencias de cada una de las líneas de la red trifásica correspondiente a la información que este entrando al concentrador en ese momento. Para el usuario este proceso ocurre de manera transparente y en ningún momento se percata de la operación el usuario solo debe iniciar el programa y establecer el puerto de conexión, esta parte de la aplicación se muestra en la figura 53.

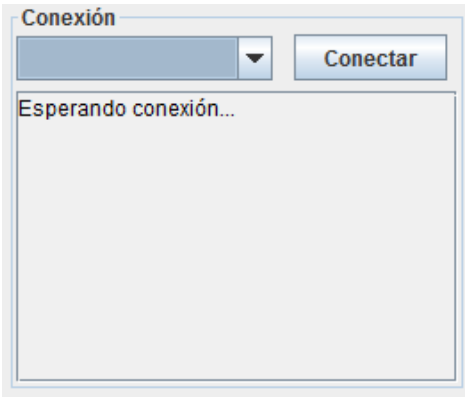


Figura 53 Interfaz de Puerto de conexión con la red

2. Consulta.

Para la consulta de la información primero se debe seleccionar el nodo al cual se le desea conocer el comportamiento de alguna de sus variables. La interfaz gráfica del software permite realizar la consulta y presentarla con su representación gráfica junto con la tabla de información esto se puede ver en la figura 54. Los tipos de consulta que se pueden realizar son los siguientes:

- Año: Permite seleccionar el año en el que se desea realizar la consulta y seleccionar cualquiera de las variables de la red, los datos gráficos corresponden al promedio de cada uno de los meses, pero los datos que se presentan en la tabla representan cada una de las mediciones que se realizaron en ese año.
- Mes: Se permite la selección del año y del mes del cual se quiera consultar, además de la variable que se quiere observar de la red, la gráfica que se presenta es el promedio diario de la variable seleccionada, la información que se muestra en la tabla corresponde a todas las mediciones.

- Día: permite la selección del año, del mes y del día, de la variable que se desea consultar, la gráfica que se presenta representa el promedio de las mediciones realizadas en una hora de medición, de igual manera la tabla presenta todo el contenido de las variables en ese día.

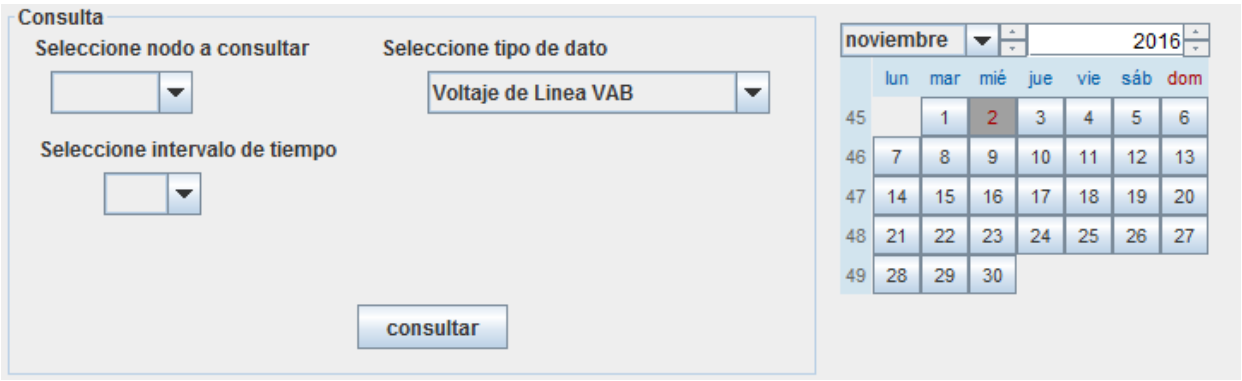


Figura 54 Interfaz de consulta

3. Petición.

El software desarrollado presenta una herramienta de petición, esto permite que atreves de la interfaz se logre realizar el envío de una petición a cualquier nodo sensor, el cual responderá con el envío de toda la información de la red con la marca de tiempo correspondiente al instante en que la orden sea recibida, esto se puede ver en la figura 55.

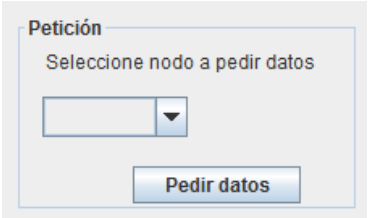


Figura 55 Interfaz de petición.

4. Visualización de información.

La información que se consulta se entrega al usuario de manera visual mediante una tabla y una gráfica que se ajustan de manera automática a la consulta realizada, la interfaz de visualización se aprecia en la figura 56 y la interfaz completa del programa en la figura 57.

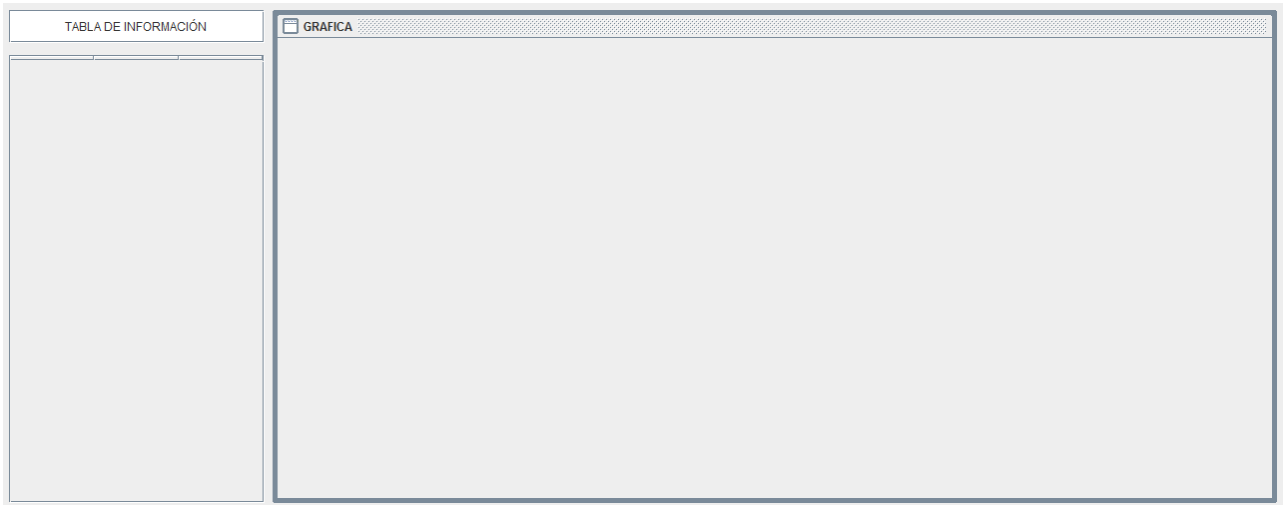


Figura 56 Interfaz de visualización.

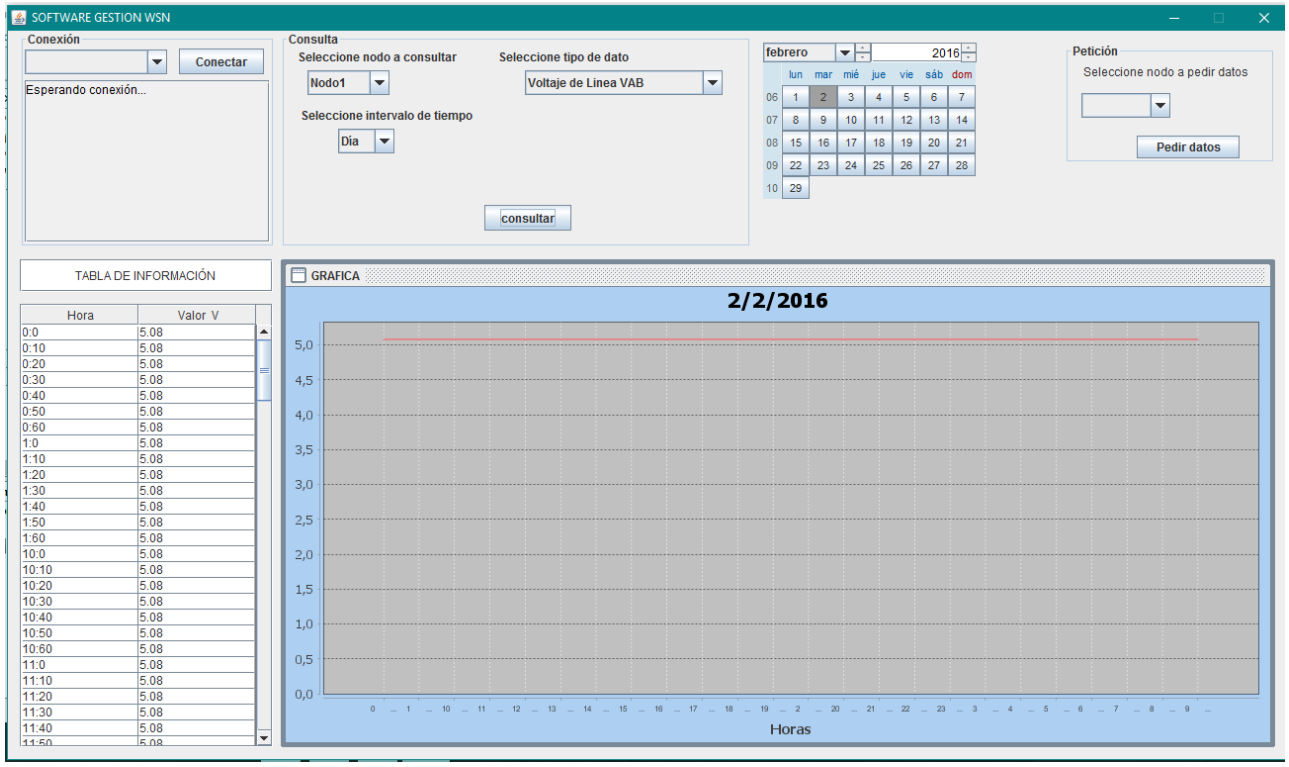


Figura 57 Interfaz completa de la aplicación.

El diagrama de flujo de la figura 58, corresponde al diseño del software del nodo concentrador, el diagrama de clases se presenta en el anexo 11.4.

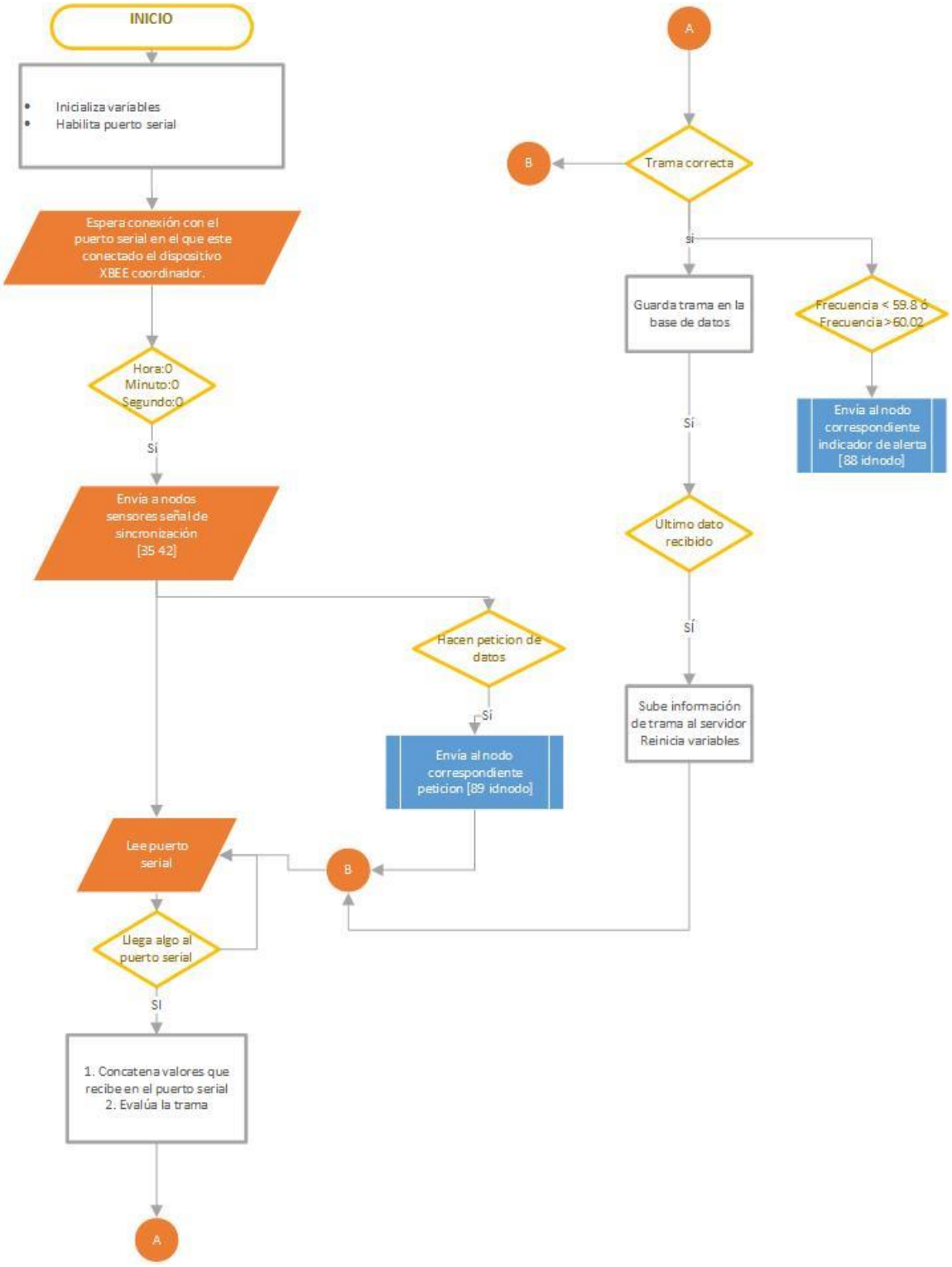


Figura 58 Diagrama de flujo del software del nodo concentrador.

5.7. Diseño de circuitos impresos.

Para la elaboración de los circuitos impresos se organizó tres tarjetas que contiene todo el esquema necesario para la elaboración del prototipo de un nodo sensor, de la siguiente manera:

- Tarjeta de sensado
- Tarjeta de adquisición de información y comunicación
- Tarjeta de alimentación.

5.7.1. Tarjeta de sensado

En esta tarjeta se disponen los tres sensores de tensión y tres sensores de corriente necesarios para la medición de la red trifásica, cada una de las fases tiene asignado un punto de salida y otro de entrada, de igual manera para la medición de la corriente de neutro se adiciona un sensor que se encargue de esto. En la figura 59 se presenta el diseño del circuito impreso realizado para esta tarjeta y en la figura 60 se muestra una perspectiva simulada tridimensional de la tarjeta diseñada.

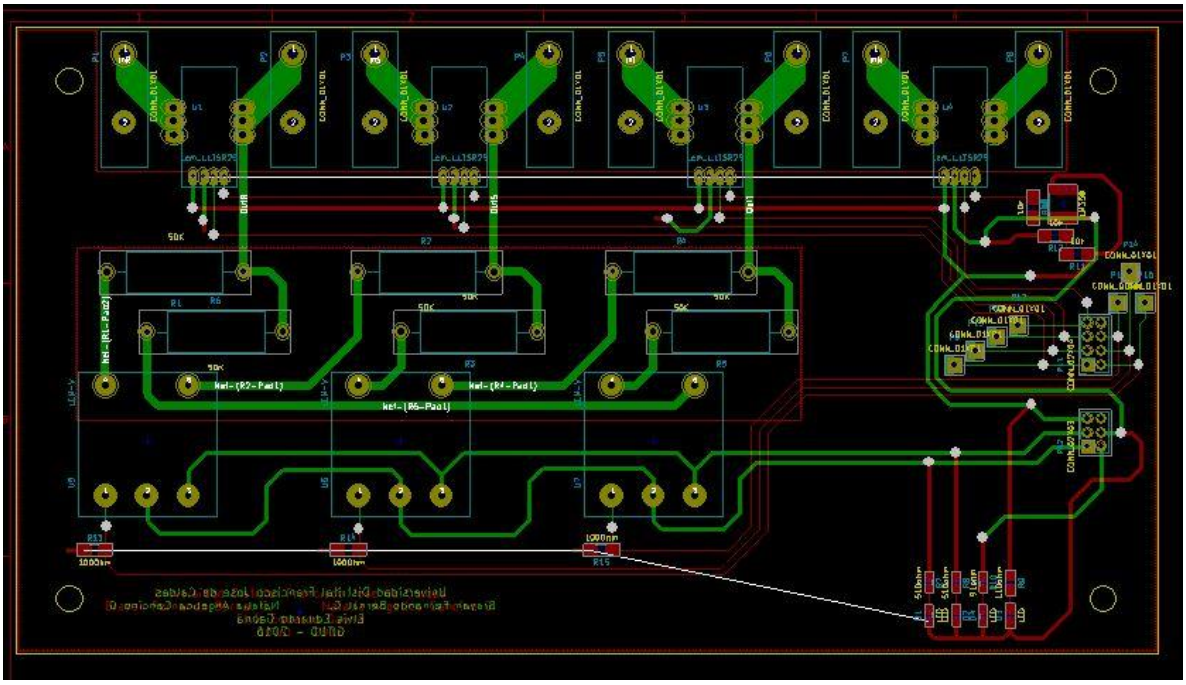


Figura 59 Diseño del circuito impreso de la tarjeta de sensado.

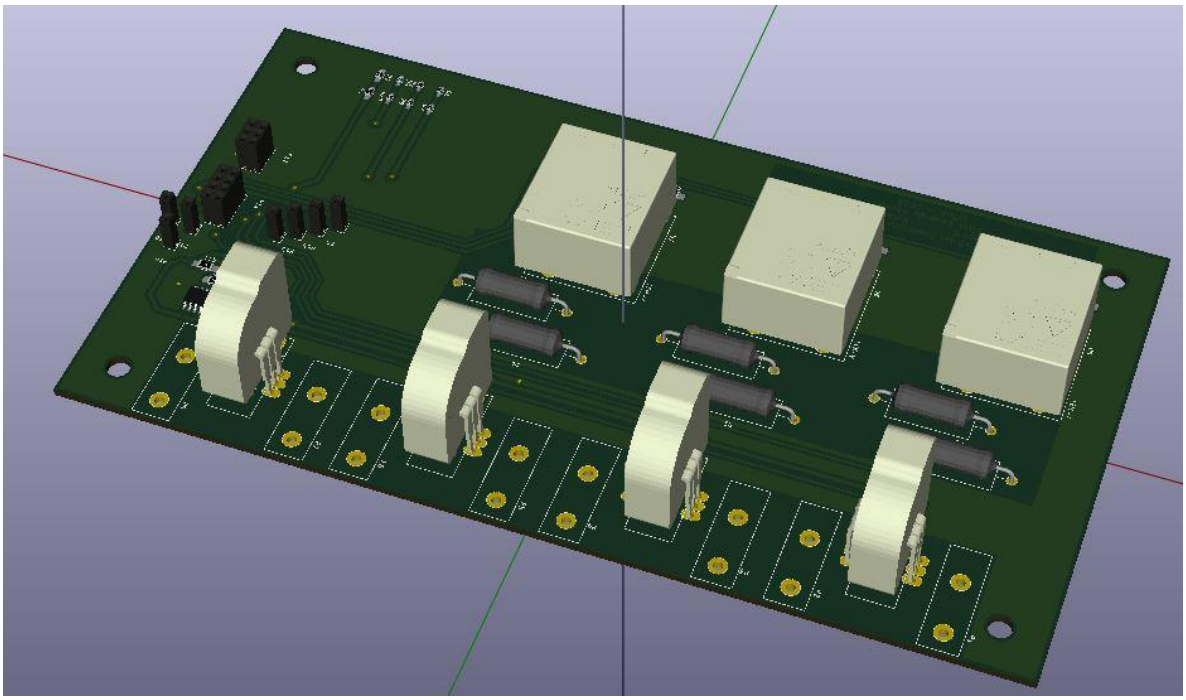


Figura 60 Tarjeta de sensado vista 3D.

5.7.2. Tarjeta de adquisición de información y comunicación

Esta tarjeta tiene como entrada las señales generadas por los sensores, esta tarjeta contiene todos los circuitos de acoples que son necesarios, de igual manera se incluye la tarjeta de procesamiento para la obtención de las variables y el módulo de comunicación Xbee. En la figura 61 se presenta el circuito impreso diseñado para esta tarjeta, en la figura 62 se muestra una perspectiva simulada tridimensional de la tarjeta diseñada.

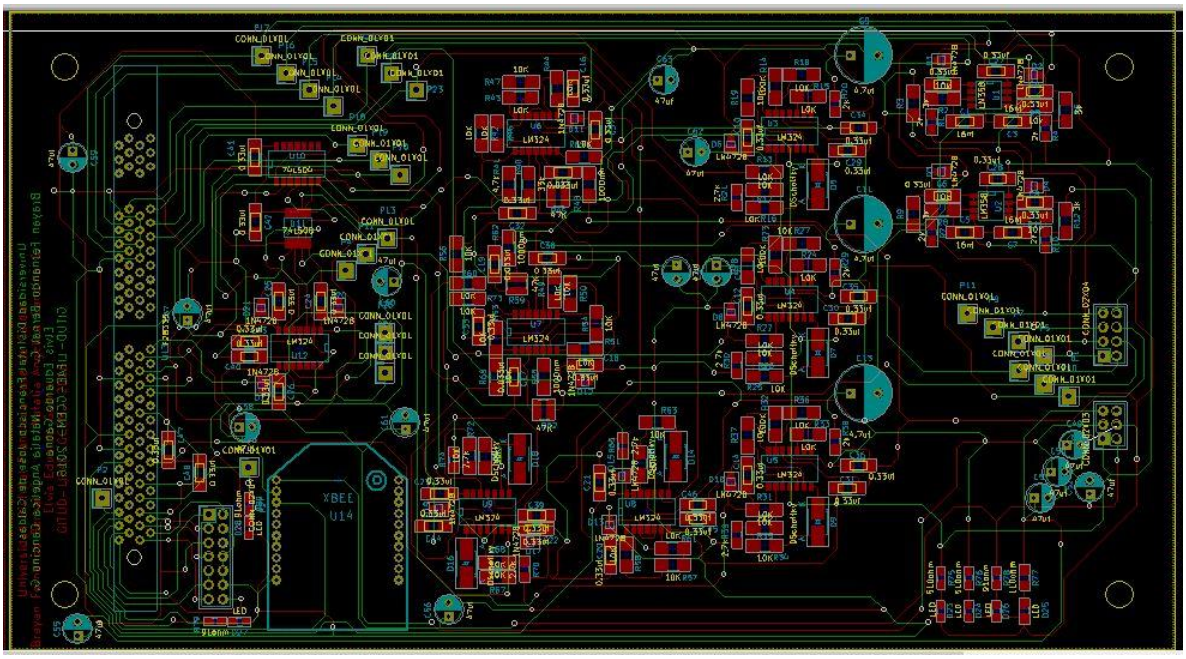


Figura 61 Diseño del circuito impreso de la tarjeta de adquisición de información y comunicación.

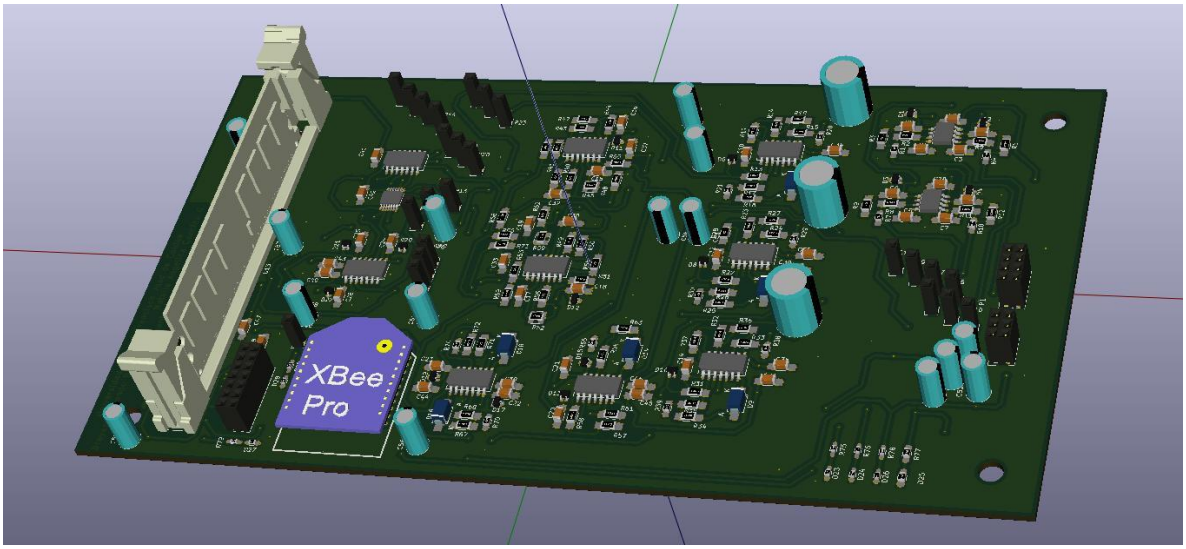


Figura 62 Tarjeta de adquisición de información y comunicación vista 3D.

5.7.3. Tarjeta de alimentación.

El diseño del prototipo del nodo sensor requiere una fuente de alimentación para todos los dispositivos que conforman el nodo. Para ello se emplea la fuente de alimentación del fabricante TRACO POWER modelo TMP 30212C, este dispositivo se alimenta de la red eléctrica y entrega dos salidas de tensión una de *+12V* y otra de *-12V*, a partir de estas señales se obtienen las fuentes adicionales que necesita el circuito que son de *+5V* y *3.3V*. Se presenta el circuito impreso diseñado para esta tarjeta en la figura 63, en la figura 64 se muestra una

perspectiva simulada tridimensional de la tarjeta diseñada, en el anexo 11.3 se presenta el diagrama del circuito completo de la tarjeta de alimentación.

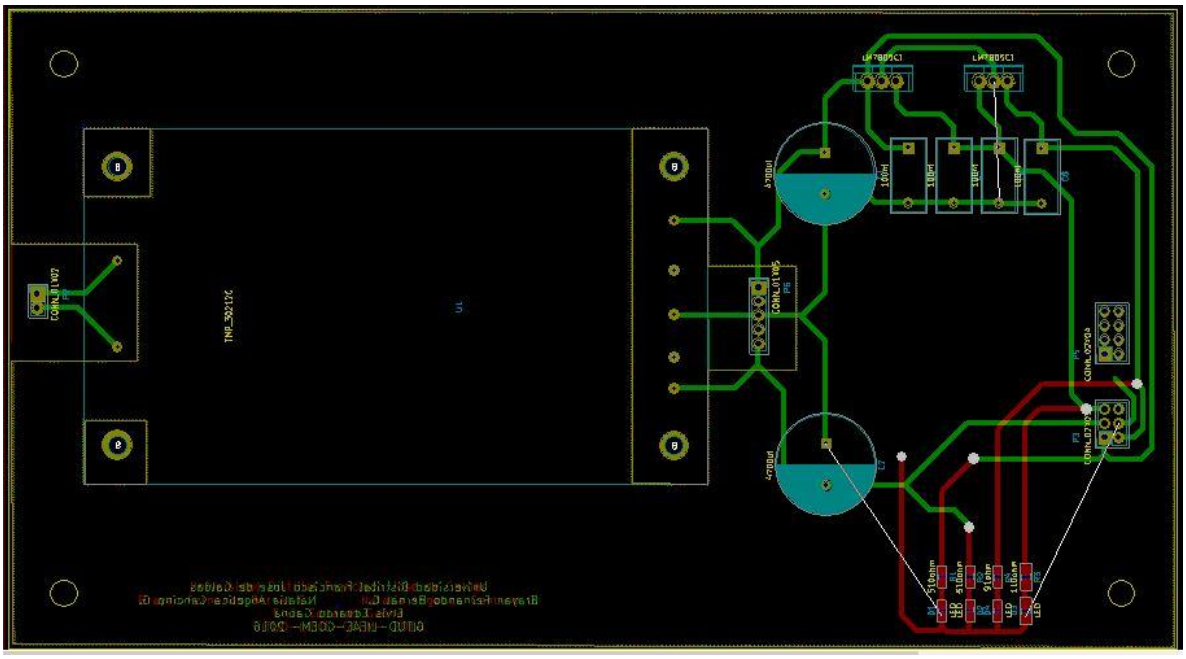


Figura 63 Diseño del circuito impreso de la tarjeta de alimentación.

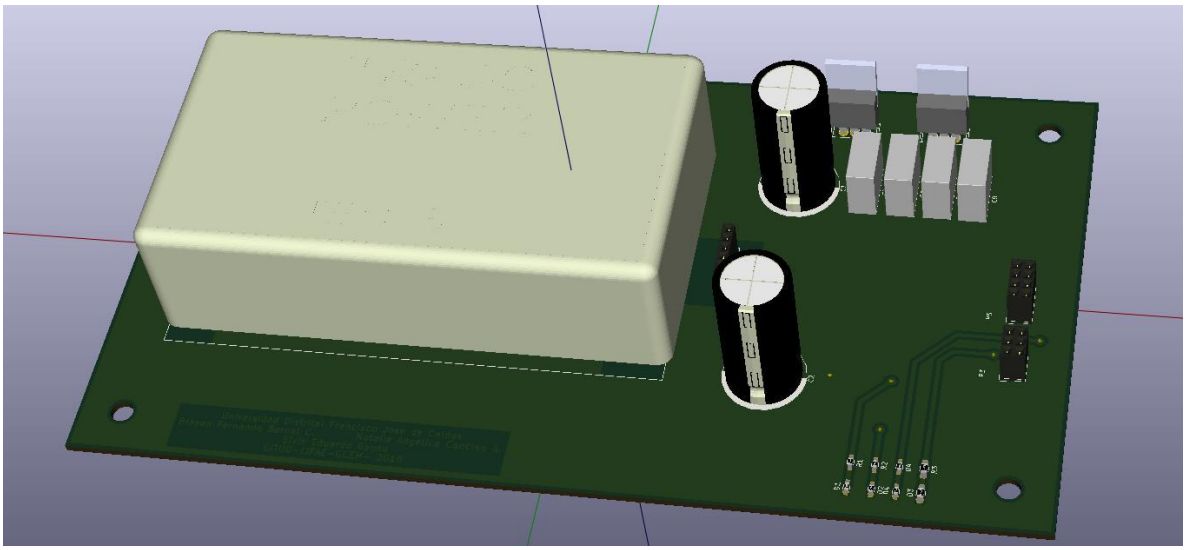


Figura 64 Tarjeta de alimentación vista 3D.

6
PRUEBAS Y RESULTADOS.

En este capítulo se muestran los resultados de las diferentes pruebas realizadas durante la elaboración del proyecto. Con base en los diseños mostrados en el capítulo anterior se realizó la implementación del sistema. Estas pruebas permitieron detectar algunos problemas en el funcionamiento que fueron corregidos.

Para el desarrollo de las pruebas se empleó un computar Lenovo, con un procesador Intel Core i5 de 1.70 GHz y una memoria RAM de 4GB.

Inicialmente las pruebas fueron realizadas de manera modular por cada una de las etapas en que fue diseñado, una vez se corrigieron los errores existentes se realizó la prueba conjunto del sistema para validar así, la red de sensores inalámbrica para microrredes.

6.1. MODULO DE SENSADO.

Este módulo está compuesto el esquema de sensores y los elementos externos necesarios para su funcionamiento indicado en los capítulos anteriores, el montaje de estos elementos se realiza en protoboard. El diagrama de conexión fue el presentado en la figura 20, para el desarrollo de las pruebas el bloque de Generación o Carga que se presenta en la figura 20, estará compuesto por el módulo de LEYBOLD 725702 presentado en la figura 65, la cual permite la variación de tensión de las líneas.

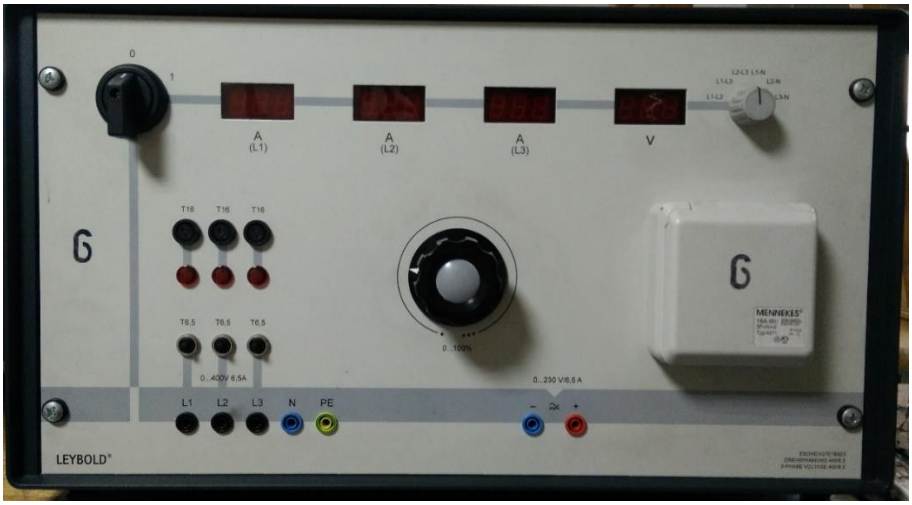


Figura 65 Fuente trifásica regulada LEYBOLD 725702

Para la variación de la corriente se dispusieron distintos reóstatos de 10 ohm de la marca PHYWE en cada una de las líneas como se muestran en la figura 66.



Figura 66 Reóstatos de 10 ohm PHYWE

Variando los valores de tensión, corriente y observando las señales entregadas por los sensores se puede observar el correcto funcionamiento de los dispositivos de sensado como se presenta en la figura 67.

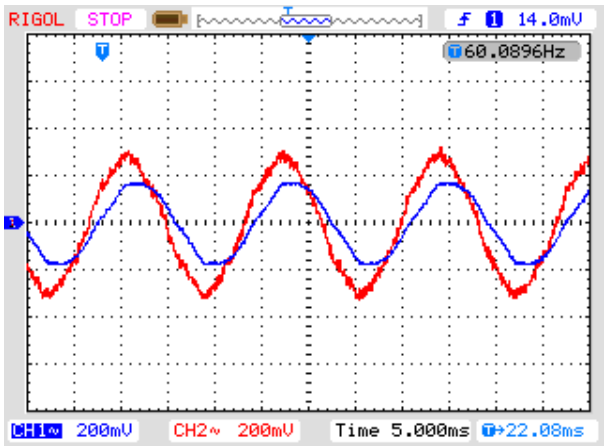


Figura 67 Señales de sensado, para una tensión de 85V y una corriente de 4A.

La respuesta del sensor de tensión se puede observar en la figura 68 y en la tabla 8 donde se evidencia el comportamiento lineal del sensor de tensión.

No.	Tensión Entrada (V)	Valor sensado (mV)		
		min	max	Vpp
1	11	-40	54	94
2	20	-72	84	156
3	30	-108	110	218
4	40	-148	156	304
5	50	-180	180	360
6	60	-216	220	436
7	70	-248	256	504
8	80	-288	290	578
9	90	-324	330	654
10	100	-360	364	724

Tabla 8 Salida Sensor de tensión.

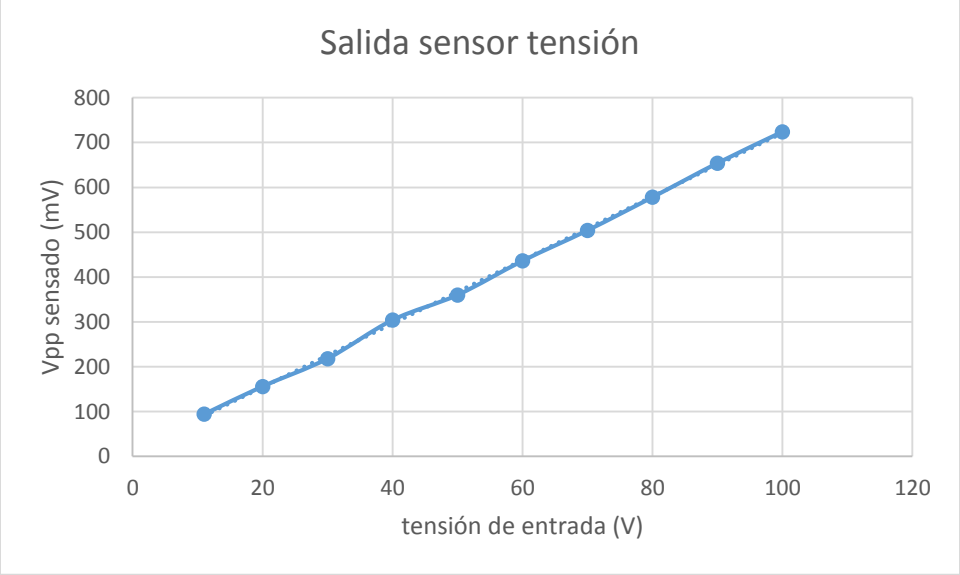


Figura 68 Comportamiento lineal del Sensor de tensión.

La respuesta del sensor de corriente se puede observar en la figura 69 y en la tabla 9 se evidencia el comportamiento lineal del sensor.

No.	Corriente Entrada (A)	Valor sensado (V)		
		min	max	Vpp
1	1	2,444	2,556	0,112
2	2	2,404	2,596	0,192
3	3	2,37	2,636	0,266
4	4	2,332	2,668	0,336
5	5	2,3	2,7	0,4

Tabla 9 Salida Sensor de Corriente.

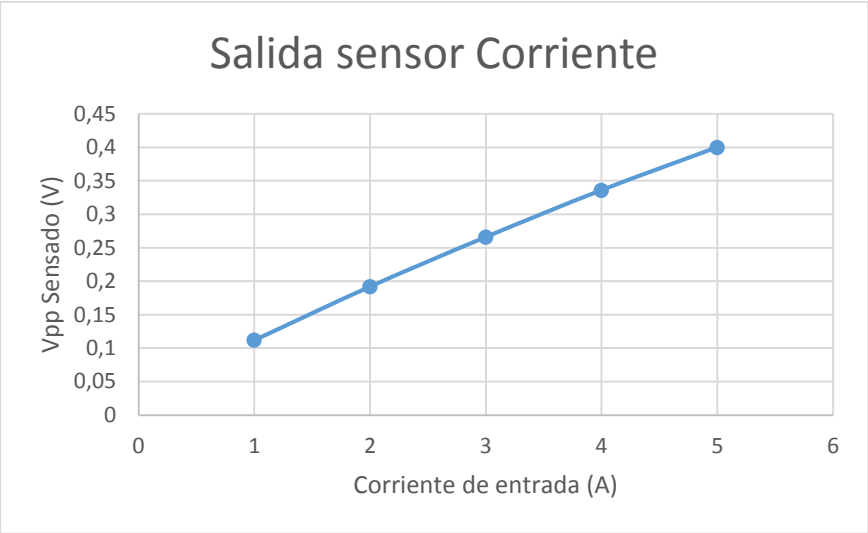


Figura 69 Comportamiento lineal del Sensor de corriente.

El comportamiento de los sensores entre si es muy similar, debido a esto solo se presentan los datos correspondientes a uno de los sensores de todos los dispuestos en el diagrama. En la figura 70 se muestra el montaje del módulo de sensado, de igual manera se presentan pequeñas variaciones que se deben a distintas causas, como:

- La variación de tensión entregada por la fuente de poder, la fuente LEYBOLD 725702 presenta pequeñas variaciones en cada una de sus líneas y además puede llegar a variar la tensión entre ellas de hasta ±2V.
- El montaje se implementó en protoboard, este es un elemento de propósito general y tiene limitantes de corriente tensión entre otras, lo cual contribuye a que se presenten pequeñas perturbaciones en el sensado.

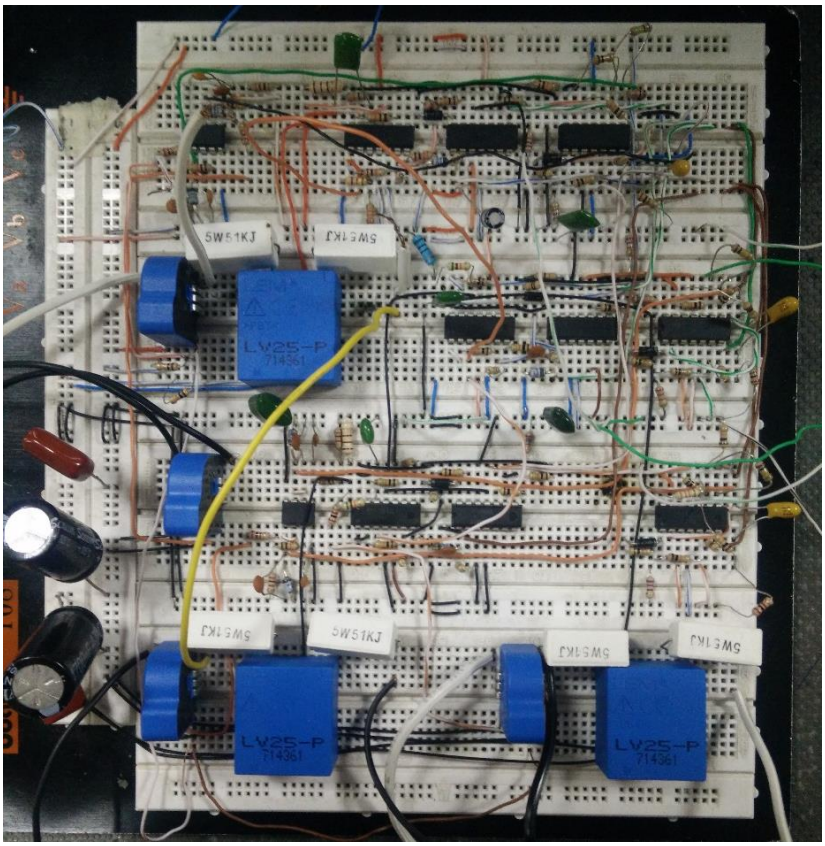


Figura 70 Montaje de los circuitos de sensado y acople.

6.2. SIMULACIONES

Se realizaron las simulaciones de cada uno de los acoples diseñados en el software de Multisim 12.0, las señales de entrada que corresponden a las salidas de los sensores se emularon con la implementación del generador de señales del software y con el principio de funcionamiento de los sensores.

6.2.1. Acople de voltaje.

En la figura 71 se presenta el resultado del circuito de acople simulado, en donde en color azul se presenta la señal emulada que hace el papel de salida del sensor de tensión, en color rojo se muestra la señal acoplada en donde se puede evidenciar el correcto funcionamiento del circuito diseñado acoplando la señal a las necesidades requeridas.

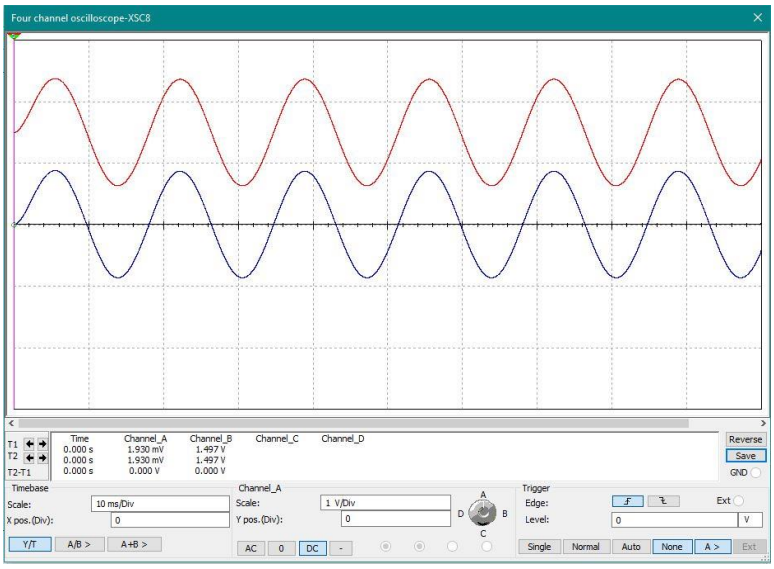


Figura 71 Simulación del acople de tensión, azul señal simulada del sensor de tensión, rojo señal de salida acoplada.

6.2.2. Acople de frecuencia.

En la figura 72 se presenta el resultado de la simulación del circuito de acople de frecuencia simulado, se puede observar en color rojo la señal de PWM generada que tiene la misma frecuencia que la señal de tensión, en color azul se observa la señal simulada del sensor de tensión, en esta grafica se evidencia que se cumple a relación de tiempo en alto y tiempo en bajo de la señal PWM respecto a la señal de entrada.

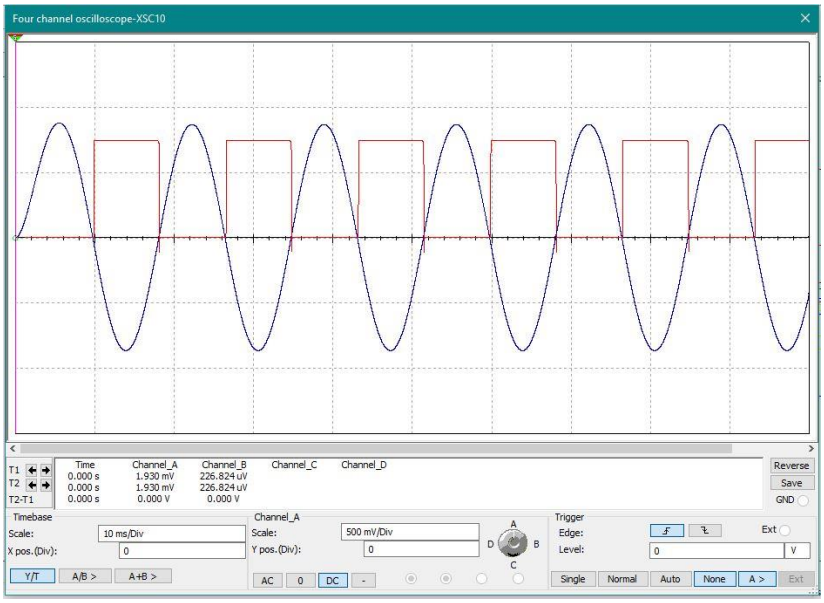


Figura 72 Simulación del acople de frecuencia, azul señal simulada del sensor de tensión, rojo señal de PWM generada.

6.2.3. Acople de corriente.

En la figura 73 se presenta el resultado de la simulación del circuito de acople de corriente, en color azul se observa la señal del sensor de corriente, en rojo se presenta la señal acoplada en donde se puede observar el correcto comportamiento del acople según los requerimientos con los que se diseñó.

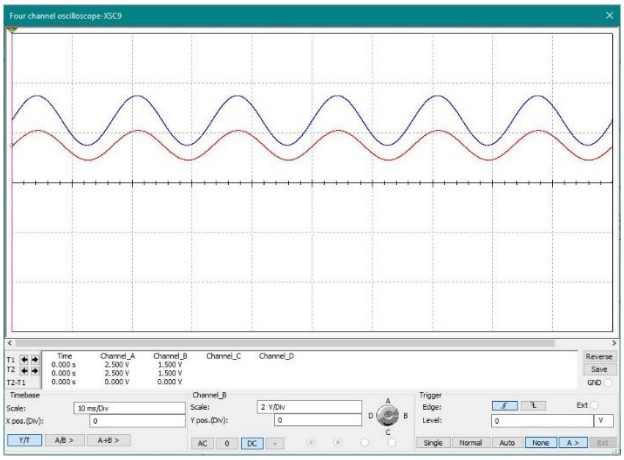


Figura 73 Simulación acople de corriente, azul señal de entrada, rojo señal acoplado.

6.2.4. Acople de desplazamiento entre la señal de tensión y corriente.

En la figura 74 se presenta el resultado de la simulación del acople de desplazamiento entre la señal de tensión y corriente, en azul se presenta la señal de PWM que se obtiene a partir de la señal que entrega en sensor de corriente, en rojo se presenta la señal de PWM que se obtiene del sensor de tensión que es la misma señal del acople de frecuencia, mediante el circuito de acople se extrae el desplazamiento que se genera entre las señal donde el tiempo en color amarillo en donde el tiempo en alto de este PWM generado equivale a el desplazamiento, se evidencia el correcto funcionamiento del circuito de acople diseñado.

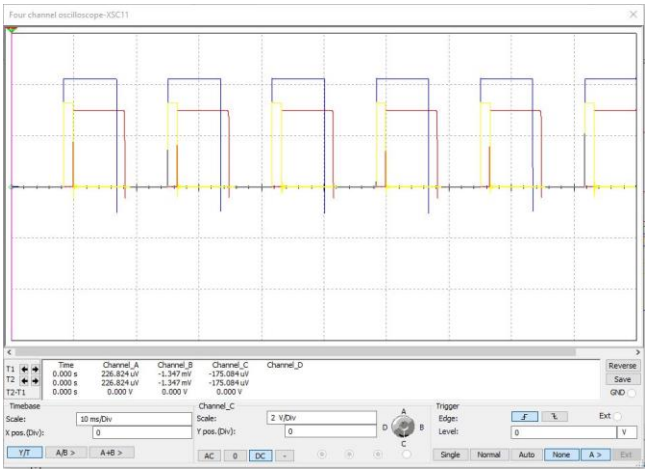
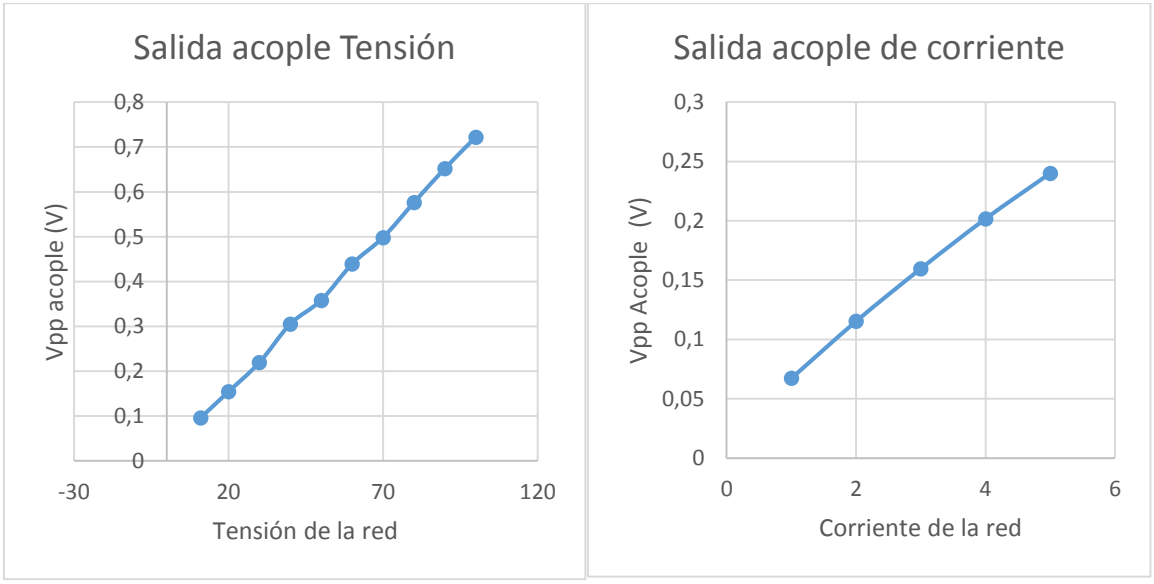


Figura 74 Simulación del circuito de acople del desplazamiento de la señal de tensión y corriente.

6.3.MODULO DE ACONDICIONAMIENTO.

Para evaluar el aspecto funcional del circuito de acople de las señales sensadas, se observó que se mantenía la linealidad de los dispositivos de medición y que las características con las cuales fueron diseñados en las distintas etapas que componen cada uno de los acoples. Cada uno de los acoples diseñados se montaron en protoboard para la realización de las pruebas necesarias que validen su funcionamiento como se observa en la figura 70, al emplear este elemento se presentaron pequeñas variaciones como igual que en las pruebas de los sensores.

En la gráfica de la figura 75a se puede observar la linealidad de la salida del circuito de acople para el sensor de tensión, en la tabla 10 se puede se evidencia el ajuste del nivel DC de la señal, así se valida el ajuste de la señal para que pueda ingresar a la tarjeta de procesamiento. Para el sensor de corriente se tienen resultados igual de favorables, se puede ver en la figura 75b y en la tabla 11.



a *b*

Figura 75 a. Señal Acoplada Tensión b. Señal Acoplada Corriente

No.	Tensión (V)	Valor Acople (V)		
		min	max	Vpp
1	11	1,46	1,556	0,096
2	20	1,428	1,582	0,154
3	30	1,392	1,611	0,219
4	40	1,355	1,66	0,305
5	50	1,32	1,678	0,358
6	60	1,284	1,723	0,439
7	70	1,252	1,75	0,498
8	80	1,212	1,788	0,576

9	90	1,176	1,828	0,652
10	100	1,14	1,862	0,722

Tabla 10 Salida Acople de tensión.

No.	Corriente Entrada (A)	Valor sensado (V)		
		min	max	Vpp
1	1	1,4664	1,5336	0,0672
2	2	1,4424	1,5576	0,1152
3	3	1,422	1,5816	0,1596
4	4	1,3992	1,6008	0,2016
5	5	1,38	1,62	0,24

Tabla 11 Salida Acople de corriente.

El resultado del circuito de acople para generar la señal de frecuencia a sensar se presenta en la figura 76, en color rojo se puede apreciar las características de la señal que resultan válidas para el cálculo de la frecuencia. Para la obtención del valor de desplazamiento entre la corriente y tensión es necesario generar una señal de PWM de la corriente, esta se puede observar en la figura 58 de color azul.

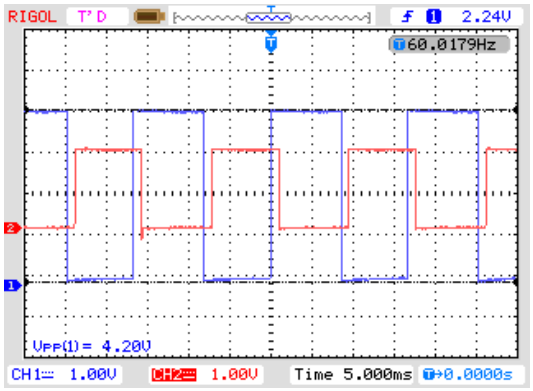


Figura 76 Señal de frecuencia, rojo frecuencia señal de tensión, azul PWM de corriente desplazada.

Para observar la distancia de desplazamiento entre la señal de tensión y de corriente se empleó el módulo de cargas inductancias IGT serie 078 que se observa en la figura 77, se realizaron variaciones en la carga y se evidenciaron las distintas variaciones en el desplazamiento de la corriente. En la figura 76 se puede observar en la señal de corriente y tensión con el desplazamiento provocado por la carga y en la figura 78 se muestra la señal de PWM donde su ciclo en alto corresponde a esa distancia de desplazamiento entre la corriente y la tensión que será calculada en el módulo de procesamiento.



Figura 77 Módulo de cargas inductancias IGT serie 078

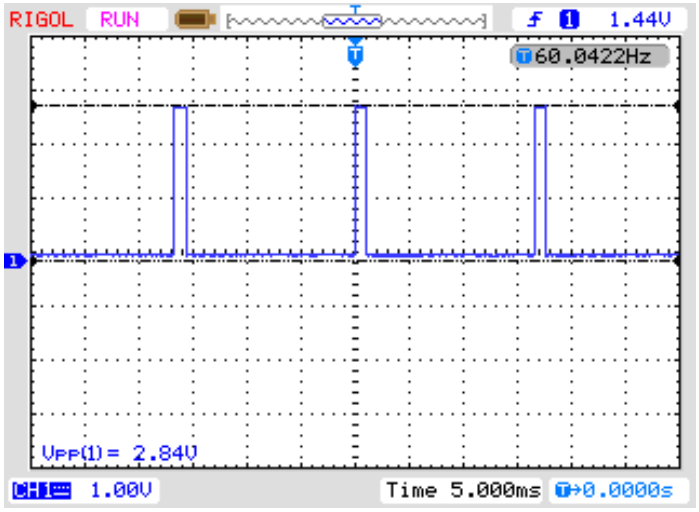


Figura 78 Señal de PWM correspondiente a la distancia de desplazamiento.

La resolución de medición de los sensores y por las características fiscas del montaje, se hace más complejo realizar mediciones de pequeñas señales de tensión, corriente y otras variables de interés, puesto que el ruido presente en la señal puede llegar a confundirse con el ruido proporcionado por el circuito haciendo invalida la medición, esto ocurre cuando se desean medir señales inferiores a los 10 voltios y corriente por debajo de los 2 amperios, los valores superiores a estos límites inferiores se realiza con una mayor precisión, abarcando la zona de trabajo de la microrred de forma exitosa.

6.4.MODULO DE PROCESAMIENTO.

6.4.1. Prueba calibración de tensiones y corrientes y cálculo de valores RMS

Con base en la figura 42 presentada en el capítulo anterior, se realizó en código en CCS que se encuentra en el anexo 11.5, para la captura de tensiones de línea a línea y corrientes de línea.

Para la calibración se obtuvo varias muestras de los máximos tanto de tensiones como de corrientes convertidos a su equivalente de 0 a 3v, y luego estas se promediaron.

En las siguientes tablas, 12 13 14 y 15, se muestra los valores obtenidos:

Tensiones línea a línea de entrada	**VR-S (V)**	**VS-T (V)**	**VT-R (V)**
	99,9	99,3	99,4
Muestras de tensiones máximos línea a línea	**VR-S (V)**	**VS-T (V)**	**VT-R (V)**
1.	1,816117	1,750916	1,876923
2.	1,815385	1,749451	1,876923
3.	1,816117	1,749451	1,879121
4.	1,816850	1,750916	1,877656
5.	1,816850	1,749451	1,876923
6.	1,816117	1,750183	1,876923
Promedio	1,816239	1,750061	1,877412

Tabla 12 Resultados prueba calibracación con una tensión de entrada de 100 V

Tensiones línea a línea de entrada	**VR-S (V)**	**VS-T (V)**	**VT-R (V)**
	50,1	49,8	49,7
Muestras de tensiones máximas línea a línea	**VR-S (V)**	**VS-T (V)**	**VT-R (V)**
1.	1,653480	1,600733	1,726007
2.	1,652747	1,600000	1,725275
3.	1,653480	1,599267	1,726007
4.	1,653480	1,598535	1,727473
5.	1,652747	1,598535	1,727473
6.	1,653480	1,599267	1,726740
Promedio	1,653271	1,599372	1,726426

Tabla 13 Resultados prueba calibración con una tensión de entrada de 50v

Corrientes de línea de entrada	**IR (A)**	**IS (A)**	**IT (A)**	**IN (A)**
	4,03	4,19	4,18	1,32

Muestras de corrientes máximas de línea	**IR (A)**	**IS (A)**	**IT (A)**	**IN (A)**
1.	1,636630	1,629304	1,635165	1,538461570
2.	1,636630	1,630037	1,635898	1,538461570
3.	1,636630	1,629304	1,634432	1,539194230
4.	1,637363	1,630037	1,635165	1,539926770
5.	1,636630	1,629304	1,634432	1,538461570
6.	1,636630	1,629304	1,634432	1,539194230
Promedio	1,636630	1,629513	1,634851	1,538984881

Tabla 14 Resultados prueba calibración con una corriente de entrada de 4 A

Corrientes de línea de entrada	**IR (A)**	**IS (A)**	**IT (A)**	**IN (A)**
	2,1	2,14	2,14	0,00
Muestras de corrientes máximas de línea	**IR (A)**	**IS (A)**	**IT (A)**	**IN (A)**
1.	1,594139	1,586081	1,590476	1,514285800
2.	1,594139	1,586081	1,591941	1,513553140
3.	1,594872	1,586081	1,594872	1,514285800
4.	1,593407	1,586081	1,591209	1,513553140
5.	1,594872	1,585348	1,591209	1,513553140
6.	1,594872	1,585348	1,591209	1,513553140
Promedio	1,594349	1,585767	1,591941	1,513762471

Tabla 15 Resultados prueba calibración con una corriente de entrada de 2 A

Con estos valores promedio obtenidos se remplazaron en las ecuaciones 5 y 7 para obtener los valores de conversión.

Con base en la figura 43, se realizó el programa que se encuentra en el anexo 11.6, donde se hizo uso de las ecuaciones de (cálculo de corriente RMS y tensión RMS). Y se obtuvieron los resultados presentados en la tabla16 y 17.

Tensiones línea a línea de entrada	**VR-S (V)**	**VS-T (V)**	**VT-R (V)**	**VR-S (V)**	**VS-T (V)**	**VT-R (V)**	**VR-S (V)**	**VS-T (V)**	**VT-R (V)**
	80, 98	81	80,99	50	50,02	50,04	35,3	35,03	35,4
Tensiones RMS	80,48	80,76	81,23	49,67	49,32	50,13	34,02	33,38	34,14

Tabla 16 Resultados prueba de tensiones RMS

Corrientes de línea de entrada	**IR (A)**	**IS (A)**	**IT (A)**	**IN (A)**	**IR (A)**	**IS (A)**	**IT (A)**	**IN (A)**	**IR (A)**	**IS (A)**	**IT (A)**	**IN (A)**
	5,03	5,03	5,01	0	4,03	4,19	4,18	0,01	3,12	3,09	3,13	0

Corrientes RMS	5,04	4,93	4,99	0,03	3,87	4,06	3,97	0,00	2.87	2,76	2,95	0,01

Tabla 17 Resultados prueba de corrientes RMS

6.4.2. Prueba de medición de frecuencias, fases y Angulo de desfase entre tensiones línea a línea y corrientes de línea

Con base en la figura 46 se realizó el código que se encuentra en el anexo 11.7, para la captura y cálculo de frecuencia fase y ángulo de desfase. En las tablas 18, 19 y 20 se evidencian los resultados obtenidos.

Frecuencia de las tensiones línea a línea de entrada	**Frec VR-S (Hz)**	**Frec VS-T (Hz)**	**Frec VT-R (Hz)**	**Frec VR-S (Hz)**	**Frec VS-T (Hz)**	**Frec VT-R (Hz)**	**Frec VR-S (Hz)**	**Frec VS-T (Hz)**	**Frec VT-R (Hz)**
	60	60	60	60	60	60	60	60	60
Frecuencia de los tensiones línea a línea calculado	59,99	59,99	59,98	60,03	60,02	60,03	60,01	60	60

Tabla 18 Resultados prueba de frecuencias

Fases de tensiones de línea a línea	**Fase VS-T**	**Fase VT-R**	**Fase VS-T**	**Fase VT-R**	**Fase VS-T**	**Fase VT-R**
	120	240	120	240	120	240
Frecuencia de los tensiones línea a línea calculado	124,45	240,49	123,91	240,54	125,53	241,63

Tabla 19 Resultados prueba de fases

Angulo de desfase entre VL vs IL de entrada	**θ *V*LR-S vs *I*LR (grados)**	**θ *V*LS-T vs ILS (grados)**	**θ VLT-R vs ILT (grados)**	**θ VLR-S vs ILR (grados)**	**θ VLS-T vs ILS (grados)**	**θ VLT-R vs ILT (grados)**	**θ VLR-S vs ILR (grados)**	**θ VLS-T vs ILS (grados)**	**θ VLT-R vs ILT (grados)**	**θ VLR-S vs ILR (grados)**	**θ VLS-T vs ILS (grados)**	**θ VLT-R vs ILT (grados)**
	55,728	29.16	55,728	41,47	59,616	34,128	29,16	45,7	27,216	36,5	37,5	43,63
Angulo de desfase entre VL vs IL calculado	55,72	29.37	55,72	41,46	59,61	34,33	29,16	44,38	27.20	35,85	36,18	42,55

Tabla 20 Resultados prueba de ángulos de desfases entre VL vs IL

6.5.MODULO DE COMUNICACIONES.

Basándose en la figura 48 se realizó el código que se encuentra en el anexo 11.8, para hacer el envío y recepción por el puerto UART y en Java se realizó el software basándose en el esquema general de procesamiento y comunicación del nodo sensor, usando como base de datos Postgresql.

En las pruebas se utilizaron transmisores XBEE Serie 1 para realizar la el enlace de comunicación de forma inalámbrica. Inicialmente se debe configurar uno de los nodos como coordinador y los otros como routers en modo transparente (AT). Para que la red funcione en topología estrella, todos los XBEE Serie 1 deben configurarse como se muestra en la tabla 21.

	XBEE Serie 1 coordinador	**XBEE Serie 1 Router 1**	**XBEE Serie 1 Router 2**
ID PAN ID	3332	3332	3332
DH Destination Address High	0	13A200	13A200
DL Destination Address Low	FFFF	409EAD94	409EAD94
CE Coordinador Enable	Coordinator	End Device	End Device
CH channel	D	D	D

Tabla 21 Configuración módulos XBEE Serie 2 topología estrella

Una de las pruebas realizadas con estos módulos consistió en enviar 15 tramas cada una de estas tramas corresponden a un tipo de variable a medir. El nodo sensor 1 envía estos datos cada 4 minutos y el nodo sensor 2 los envía 2 minutos después, ya que en el caso que se envíen los datos de forma simultánea se pueden combinar la información enviada de cada nodo y el nodo central la guardara de manera errónea. Las tramas enviadas por cana nodo sensor se muestran en la tabla 22, y como las recibe el nodo central y los almacena en la base de datos se muestra en la figura 79.

Id inicio	**Nodo**	**Tipo**	**Día**	**Mes**	**Año parte 1**	**Año parte 2**	**Hora**	**Minuto**	**Valor entero 1**	**Valor entero 2**	**Valor entero 3**	**Decimal 1**	**Decimal 2**	**Id final**
60	1	1	6	11	20	16	0	1	1	2	4	2	3	62
60	1	2	6	11	20	16	0	1	1	2	4	4	5	62
60	1	3	6	11	20	16	0	1	1	2	4	2	7	62
60	1	4	6	11	20	16	0	1	0	0	3	3	2	62
60	1	5	6	11	20	16	0	1	0	0	3	4	3	62
60	1	6	6	11	20	16	0	1	0	0	3	7	6	62

60	1	7	6	11	20	16	0	1	0	0	3	5	2	62
60	1	8	6	11	20	16	0	1	1	2	0	3	5	62
60	1	9	6	11	20	16	0	1	2	4	0	3	2	62
60	1	10	6	11	20	16	0	1	0	6	2	0	0	62
60	1	11	6	11	20	16	0	1	0	6	0	0	0	62
60	1	12	6	11	20	16	0	1	0	6	0	0	0	62
60	1	13	6	11	20	16	0	1	0	3	0	2	3	62
60	1	14	6	11	20	16	0	1	0	1	0	3	2	62
60	1	15	6	11	20	16	0	1	0	2	2	3	2	62
60	2	1	6	11	20	16	0	3	1	2	5	2	3	62
60	2	2	6	11	20	16	0	3	1	2	5	4	5	62
60	2	3	6	11	20	16	0	3	1	2	5	2	7	62
60	2	4	6	11	20	16	0	3	0	0	5	3	2	62
60	2	5	6	11	20	16	0	3	0	0	5	4	3	62
60	2	6	6	11	20	16	0	3	0	0	5	7	6	62
60	2	7	6	11	20	16	0	3	0	0	5	5	2	62
60	2	8	6	11	20	16	0	3	1	2	1	3	5	62
60	2	9	6	11	20	16	0	3	2	4	1	3	2	62
60	2	10	6	11	20	16	0	3	0	6	0	0	0	62
60	2	11	6	11	20	16	0	3	0	6	0	0	0	62
60	2	12	6	11	20	16	0	3	0	6	0	0	0	62
60	2	13	6	11	20	16	0	3	0	0	3	2	3	62
60	2	14	6	11	20	16	0	3	0	0	3	3	2	62
60	2	15	6	11	20	16	0	3	0	0	3	2	1	62

Tabla 22 Tramas enviadas al nodo concentrador

nodo character(15)	tipo character(15)	dia numeric	mes numeric	año numeric	hora numeric	min numeric	valor double precision
1	1	6	11	2016	0	1	124.23
1	2	6	11	2016	0	1	124.45
1	3	6	11	2016	0	1	124.27
1	4	6	11	2016	0	1	3.32
1	5	6	11	2016	0	1	3.43
1	6	6	11	2016	0	1	3.76
1	7	6	11	2016	0	1	3.52
1	8	6	11	2016	0	1	120.35
1	9	6	11	2016	0	1	240.32
1	10	6	11	2016	0	1	62
1	11	6	11	2016	0	1	60
1	12	6	11	2016	0	1	60
1	13	6	11	2016	0	1	30.23
1	14	6	11	2016	0	1	10.32
1	15	6	11	2016	0	1	22.32
1	16	6	11	2016	0	1	356.355932194192
1	17	6	11	2016	0	1	419.957943152951
1	18	6	11	2016	0	1	459.696209021199
1	19	6	11	2016	0	1	412.4436
1	20	6	11	2016	0	1	426.8635
1	21	6	11	2016	0	1	467.2552
1	22	6	11	2016	0	1	207.653973645988
1	23	6	11	2016	0	1	76.4707369847631
1	24	6	11	2016	0	1	177.453757233858
2	1	6	11	2016	0	3	125.23
2	2	6	11	2016	0	3	125.45
2	3	6	11	2016	0	3	125.27
2	4	6	11	2016	0	3	5.32
2	5	6	11	2016	0	3	5.43
2	6	6	11	2016	0	3	5.76
2	7	6	11	2016	0	3	5.52
2	8	6	11	2016	0	3	121.35
2	9	6	11	2016	0	3	241.32
2	10	6	11	2016	0	3	60
2	11	6	11	2016	0	3	60
2	12	6	11	2016	0	3	60
2	13	6	11	2016	0	3	3.23
2	14	6	11	2016	0	3	3.32
2	15	6	11	2016	0	3	3.32
2	16	6	11	2016	0	3	665.165236687741
2	17	6	11	2016	0	3	680.050227169831
2	18	6	11	2016	0	3	720.344186601271
2	19	6	11	2016	0	3	666.2236
2	20	6	11	2016	0	3	681.1935
2	21	6	11	2016	0	3	721.5552
2	22	6	11	2016	0	3	37.5378888471626
2	23	6	11	2016	0	3	39.4496257081309
2	24	6	11	2016	0	3	41.7870730823995

Figura 79 Tramas recibidas y almacenadas

Para otra prueba del funcionamiento de transmisión y recepción del sistema, se realizó variando la distancia entre los nodos. Las tramas enviadas por el nodo sensor son valores predeterminados ya que para esta prueba no se tenía acceso a los equipos del laboratorio y en este caso solo se probó con un nodo sensor, como se ilustra en la figura 80.



Figura 80 Prueba realizada a 13 m

La tabla 23 muestra el tiempo que tardan en llegar las 15 tramas correctamente al nodo concentrador a diferentes distancias con línea de vista y sin línea de vista.

Distancia (m)	Tiempo de llegada (s)	línea de vista
14	6	Si
19	10	Si
28	12	Si
44	12	Si
23	18	No
13	16	No

Tabla 23 prueba de transmisión y recepción del sistema con diferentes distancias

Teniendo como base estos resultados de esta sección, se puede observar que al momento realizar la comunicación entre los nodos, los tiempos de transmisión y la pérdida de información, varían en forma considerable según las condiciones físicas del entorno entre los dos puntos. Cuando se presenta algún obstáculo entre los nodos y se inhabilita una línea de vista directa entre los dos puntos ocurren retardos en la trasmisión puesto que mucha de la

información se pierde y es necesario retrasmitirla. Esto puede causar que se presenten errores en la sincronización de los nodos o que incluso la información recolectada por los distintos nodos sensores se combine, alterando la información almacenada que corresponde al punto de la microrred que se está sensando.

6.5.1. Pruebas de calidad del enlace de comunicación.

Una prueba para conocer la calidad de alcance entre dos módulos Xbee, fue realizada con el software X-CTU usando la herramienta Range test, para la cual es necesario tener un módulo de radio local conectado a un PC o laptop y se añade a XCTU, y un dispositivo de control remoto en la misma red que el dispositivo local. El dispositivo local debe estar configurado en modo API y el local en modo AT. Este test consistió en enviar 69 bytes de un dispositivo a otro.

Una vez inicie el test en el grafico RSSI muestra el tráfico de los dispositivos remotos y locales durante la sesión de prueba de rango como se muestra en la figura 81, también muestra la cantidad y porcentaje de los paquetes enviados, recibidos y paquetes perdidos tanto como los enviados como los recibidos como se muestra en la figura 82.

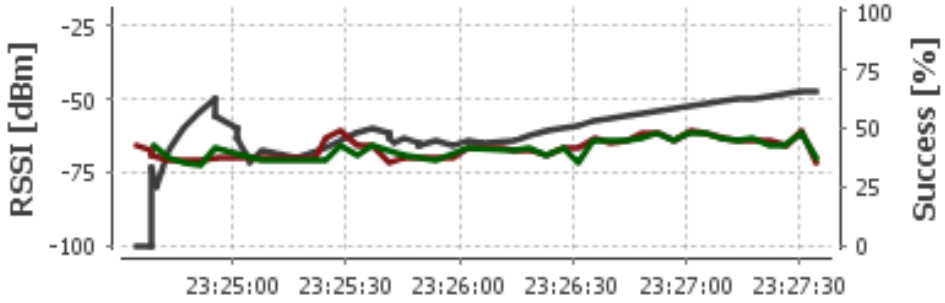


Figura 81 Grafica range test

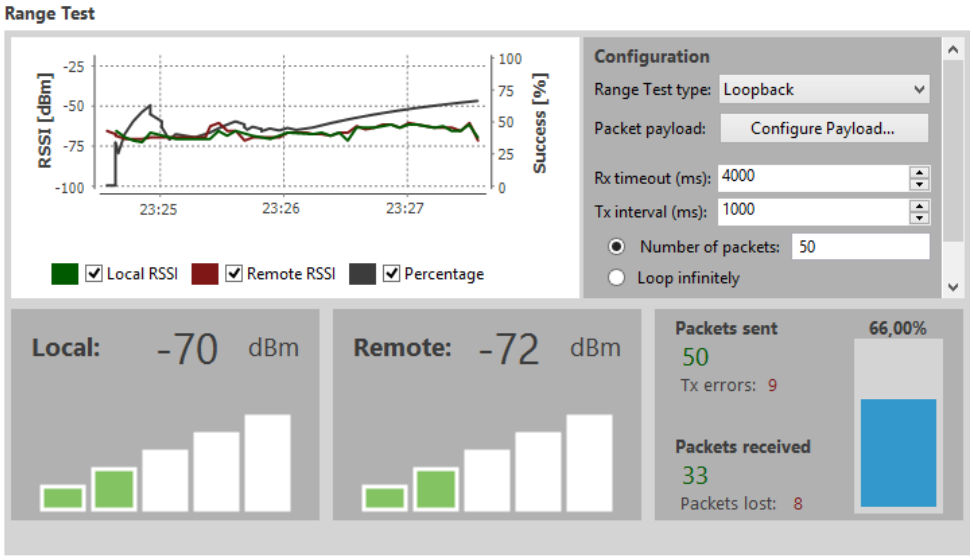


Figura 82 Range test

Para observar como varia la calidad de transmisión y recepción respecto a la distancia a la que se encuentre el dispositivo remoto al local se varió la distancia del dispositivo remoto a varias distancias y los resultados fueron los siguientes como se muestra en la tabla 24.

Distancia (m)	Paquetes Tx	Paquetes Tx perdidos	Paquetes Rx	Paquetes Rx perdidos	Calidad de Tx y Rx (%)	Línea de vista
14	50	0	49	0	100	Si
19	50	6	40	4	80	Si
28	50	9	33	8	66	Si
44	50	15	32	3	64	Si
23	50	28	12	10	24	No
13	50	20	24	6	44	No

Tabla 24 Resultados range test de los módulos Xbee

Otra prueba realizada con X-CTU fue la prueba de rendimiento de los módulos Xbee, esta prueba permite medir la velocidad de transferencia entre dos módulos de la misma red. La herramienta para hacer esta prueba se llama en el X-CTU Throughput, como se ve en la figura 83 presenta la velocidad de transferencia instantánea color azul claro y media línea de color azul oscuro, en la figura 84 se presenta la prueba de rendimiento.

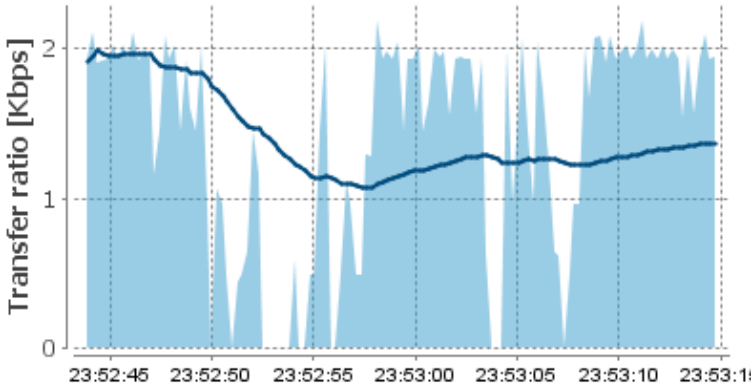


Figura 83 Grafica prueba de rendimiento

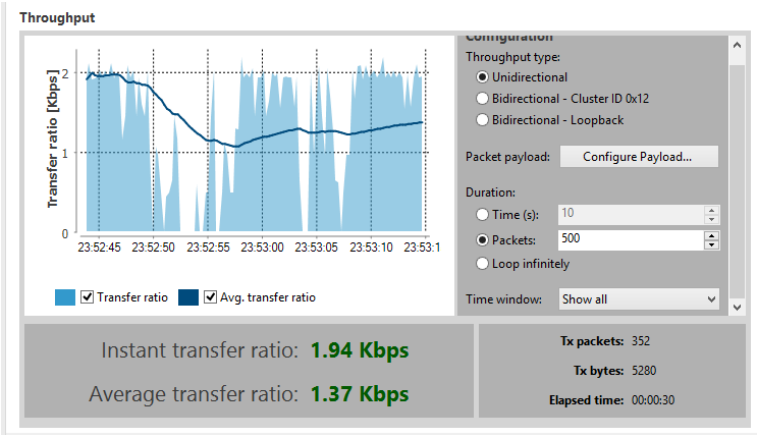


Figura 84 Prueba de rendimiento

Al igual que el range test realizado a diferentes distancias se realizó la prueba y se obtuvo los resultados de la tabla 25.

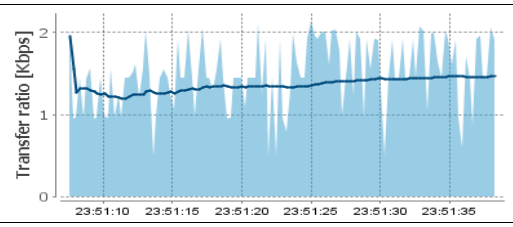
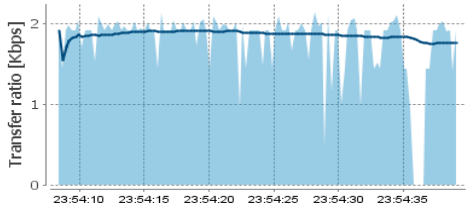
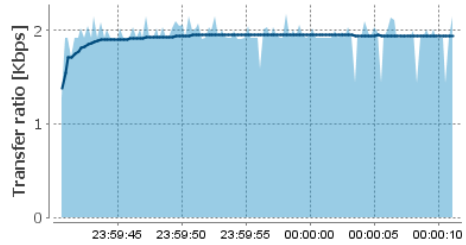
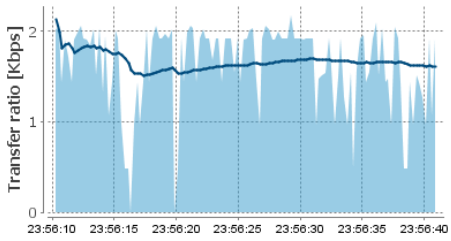
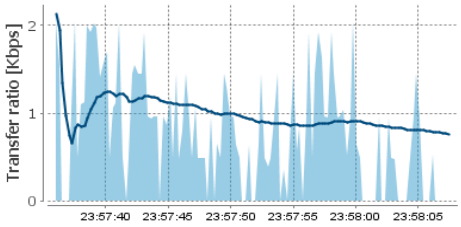
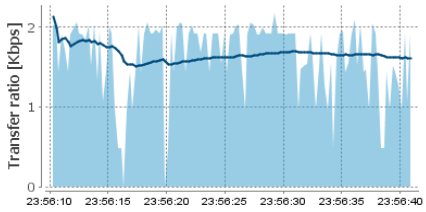
Distancia (m)	Paquetes Tx		Línea de vista
14	500		Si
19	500		Si
28	500		Si
44	500		Si
23	500		No
13	500		No

Tabla 25 Resultados prueba de rendimiento de los módulos Xbee

Con los resultados adquiridos de las pruebas de range test y rendimiento, se puede observar que a medida que aumenta las distancias entre los módulos locales y remotos Xbee .Y cuando

no existe línea de vista entre los dispositivos aumenta la perdida de datos tanto transmitidos como recibidos. Así mismo la potencia y la velocidad de transmisión disminuyen. Lo que explicaría los tiempos de llegada de la prueba de transmisión y recepción del sistema con diferentes distancias también realizadas, los tiempos en que tarda el software en recibir y reconocer una de las 15 tramas aumenta a medida que las pérdidas ya sea por distancia o elementos en el ambiente también aumentan.

6.6.PROTOTIPO NODO SENSOR Y NODO CONTROLADOR.

Se realizó la puesta en funcionamiento del prototipo de sistema de medición con los nodos sensor y nodo concentrador. Esta prueba se realizó en los laboratorios de la Facultad de Ingeniería de la Universidad Distrital Francisco José de Caldas, en la figura 85 se observa la disposición completa del montaje.

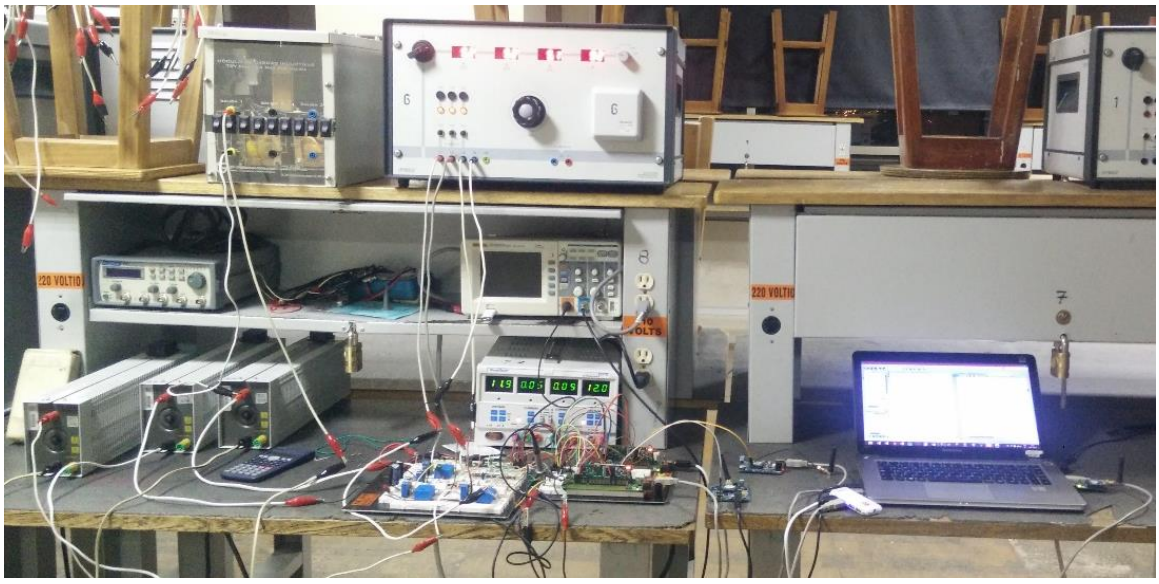


Figura 85 Montaje del prototipo.

El desarrollo de la prueba estuvo sujeta a las limitaciones técnicas de elementos de laboratorio y los materiales para el prototipo que se disponen. Por esto se comparten el sistema de sensores y de acople de las señales, para dos distintos módulos de procesamiento y de comunicación, de esta manera se obtienen dos nodos sensores que están realizando la labor de medición en una misma red eléctrica.

Los elementos necesarios para realizar la prueba son los mismos mencionados en la las pruebas del módulo de sensado sección 5.1 y de acondicionamiento sección 5.2.

Para la puesta de funcionamiento de la red de sensores se siguen los siguientes pasos:

1. Cortar el suministro de energía de la red eléctrica donde se va a realizar la medición.
2. Identificar el esquema de medición en la red trifásica.
3. Realizar la conexión a la tarjea de sensado.

4. Alimentar el nodo sensor.

Iniciar aplicativo de software y establecer conexión con el módulo de comunicación del concentrador.

6. Conectar el servicio de la red eléctrica trifásica.

7. El sistema a parir de ese punto ya empezara su puesta en marcha y la información recolectada se puede observar en la interfaz gráfica del software con ayuda del módulo de consulta del mismo.

El desarrollo de la prueba consistió en la variación de los distintos parámetros de la red, específicamente tensiones, corrientes y distancia de desfase entre estas señales, puesto que son las únicas variaciones que se permiten realizar con los elementos disponibles. Esto se realiza con la variación de las cargas de cada una de las líneas de la red trifásica mezclando el banco de inductancias con los reóstatos en diferentes configuraciones y de manera aleatoria. De igual manera por las características de la fuente trifásica se realizaron variaciones de tensión en las líneas, pero las variables de frecuencia y de fase se mantienen de manera constante sin posibilidad de ser modificadas, puesto que su variación se debe a problemas en la red lo cual para casos prácticos no es conveniente.

Estas señales entran a las dos tarjetas de procesamiento para obtener la información útil de ellas y ser transmitida al nodo concentrador, con lo cual se culmina de manera exitosa la labor del nodo sensor.

Para verificar y validar el funcionamiento en el nodo concentrador se utiliza la herramienta Netbeans, por medio de la consola se imprime cuando las tramas llegan correctamente y son guardadas en la base de datos, también se imprime en el caso que las tramas no llegan correctamente esto ocurre para la información que llega de manera automática o es solicitada por medio de la aplicación, esto se puede observar en la figura 86.

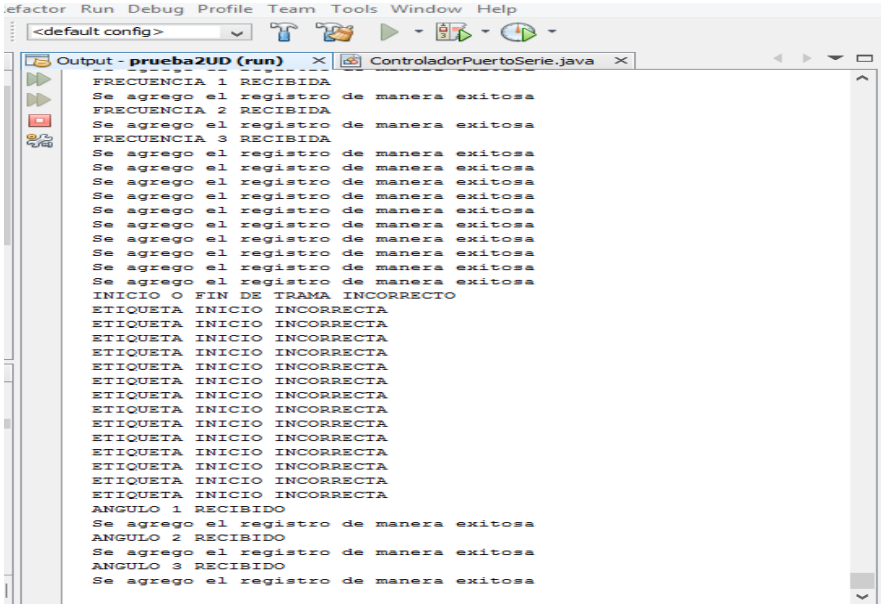


Figura 86 Consola de Netbeans

Parte de la información almacenada en la base de datos se puede observar en la figura 79, y por medio de la aplicación desarrollada se puede ver la información de manera gráfica, en la figura 87 se puede observar el panel completo de la aplicación mostrando la consulta realizada la presentación de la información en la tabla y de manera gráfica. En la figura 88 se ilustra el comportamiento de distintas variables de los dos nodos sensores sujetos a prueba.

El código uno de los nodos sensores para el desarrollo de esta prueba se encuentra en el anexo 11.9

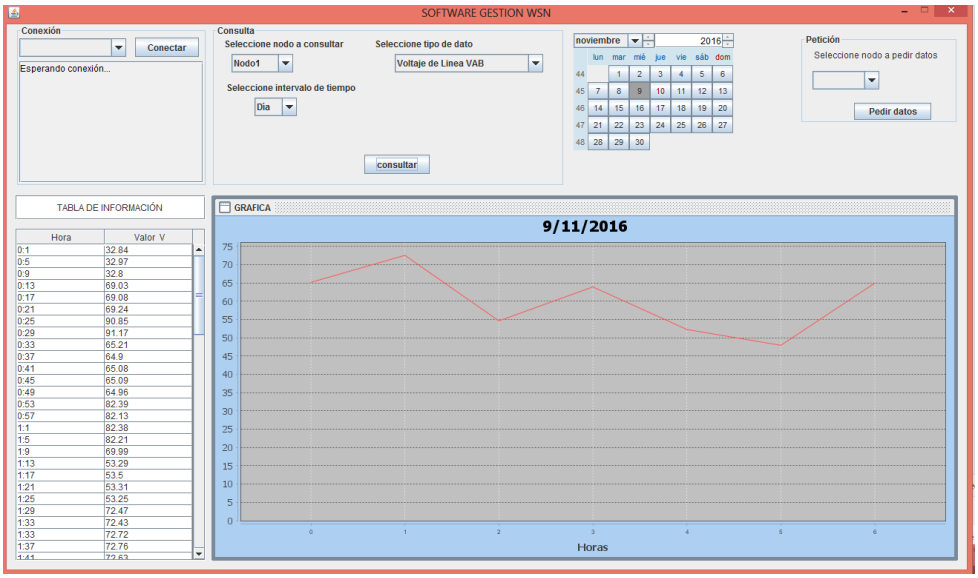


Figura 87 Panel completo del Software desarrollado.

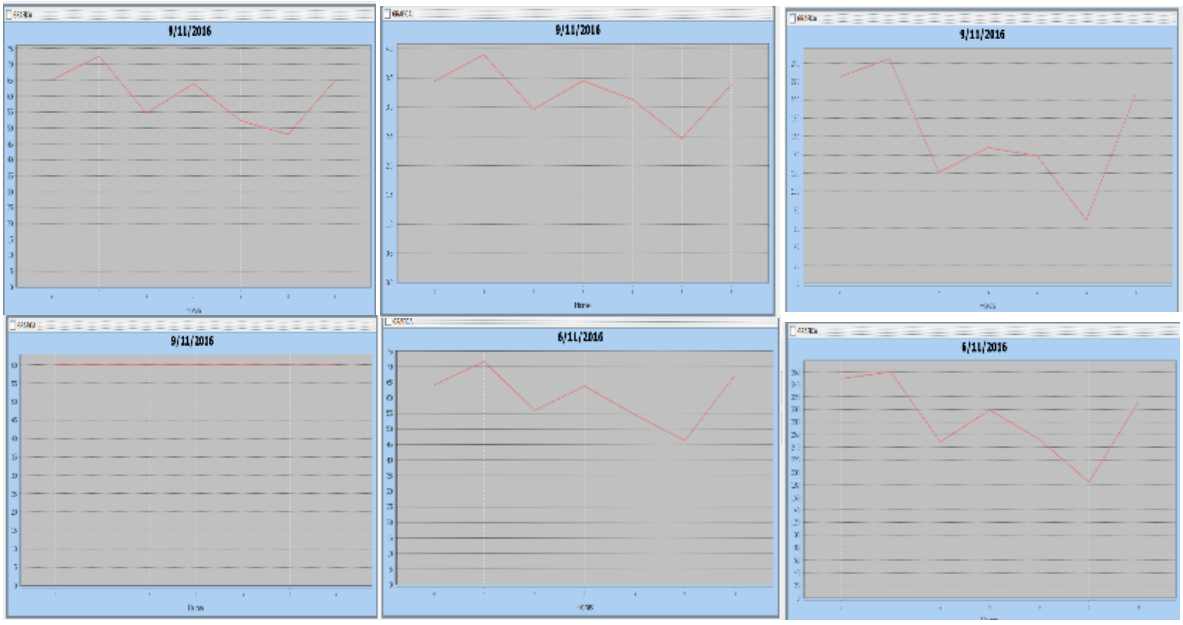


Figura 88 Graficas generadas por el software a partir de la información sensada de la microrred.

7
CONCLUSIONES.

Con la investigación realizada sobre el estado del arte de las microrredes y las redes de sensores inalámbricos que se presenta en este documento, se logró identificar los posibles elementos de software y hardware que permiten la implementación de un prototipo inalámbrico de nodo sensor y nodo concentrador.

La implementación de este proyecto permite el fortalecimiento y el desarrollo de las redes de sensado inalámbrico, brindando una herramienta de medición con la capacidad de adquirir información que permite conocer el comportamiento de la red eléctrica trifásica de una microrred.

Se evidenció que el uso de la tecnología de procesamiento brindada por la tarjeta ADSP de Texas instruments, permite facilitar la implementación y el manejo de señales analógicas y digitales, puesto que los módulos que poseen como, ADC, eCAP, SCI, Timers, entre otros, facilitan el desarrollo de ciertas tareas evitando el uso adicional de código de programación y también reduce la cantidad de componentes electrónicos necesarios para la implementación del prototipo.

Fue posible lograr el diseñó y la implementación de un dispositivo electrónico de medición, el sensado se realizó con dos tipos de sensores, uno de tensión y otro de corriente, por medio del acople y la tarjeta de procesamiento se analizó la señal entregada por los sensores y se obtuvieron las variables de tensión, corriente, frecuencia, fase y factor de potencia de la red trifásica para luego ser transmitidas, eliminando así el uso de otros sensores para la obtención de estas señales y reduciendo la cantidad de elementos que interactúan con la red eléctrica de la microrred.

Con las pruebas realizadas mediante el software XCTU, se logró establecer una relación entre la distancia, la línea de vista y la calidad en la transmisión y recepción de la red de sensores inalámbricos, de esta manera se determinó el mejor modo de operación de los transmisores Xbee serie 1 para su implementación, atendiendo de esta forma los niveles de calidad adecuados para el establecimiento del enlace de comunicación de la red inalámbrica que permita él envió de los parámetros de los distintos nodos sensores de la microrred.

Se desarrolló un aplicativo en el software NetBeans, enlazado a la herramienta PostgreSQL con la cual se permite el almacenamiento de la información adquirida por el nodo sensor y su respectiva visualización con una interfaz de fácil manejo, generando una herramienta que permite al usuario observar el comportamiento de las distintas variables medidas en los distintos puntos de la microrred donde se encuentren los nodos sensores.

8
APORTES.

El trabajo realizado representa una base para la línea de investigación de redes de sensores inalámbricos del grupo de trabajo GITUD, enfocado a las microrredes y a sistemas de medición inteligente, por lo cual la documentación que se presenta en este trabajo de grado se realiza de la manera más específica, clara, precisa y detallada con el objetivo de trasmitir los conocimientos, experiencias y resultados que se adquirieron en este proceso. Para que así, las personas que deseen trabajar en esta área puedan replicar y continuar con el trabajo realizado.

El desarrollo de este trabajo estuvo apoyado económicamente por el centro de investigación innovación y desarrollo científico de la universidad distrital CIDC, permitiendo así la generación de un libro de investigación que recopila de manera general el trabajo desarrollado en este proyecto de grado.

En el evento conocido como "Workshop on Engineering Applications" realizado entre el 28 y el 30 de octubre de 2015 por la universidad Distrital Francisco José de Caldas, la rama IEEE UD y otras entidades. Se presentó el paper titulado "Measurement infrastructure on microgrids using wireless sensors" que se elabora con los conocimientos adquiridos atreves de la revisión del estado del arte de este proyecto.

La aplicación de software desarrollada que almacena la información de la red de sensores y visualizar su comportamiento, se encuentra registrada ante la Dirección Nacional de Derechos de Autor.

El desarrollo de este proyecto permitió la elaboración un prototipo funcional de nodo sensor, con las características de diseño realizadas y presentados en este documento. Entregando así una herramienta capaz de adquirir las distintas variables de interés de una red eléctrica trifásica.

9
TRABAJOS FUTUROS.

La ampliación de la cobertura en una red de sensores inalámbricos requiere del uso de tecnologías de transmisión de datos con mejor sensibilidad en el receptor o de la implementación de diferentes topologías de red, para esto es necesario explorar el uso nuevas tecnologías y estándares con mejores características como lo son; la tasa de transferencia más alta y menor consumo de energía que mejoren la cobertura.

En cuanto al consumo de energía en los nodos sensores, es necesario mejorar el esquema de alimentación de los nodos, esto debido a que su restringido uso en ambientes aislados hace que la mejor opción sea el uso de baterías. Para lograr un mejor uso de este recurso es importante hacer uso de dispositivos de bajo consumo, tamaño y peso que permitan aprovechar las prestaciones que ofrecen como por ejemplo los modos de funcionamiento de bajo consumo y estados de hibernación en los transmisores.

En cuanto a la confiabilidad del sistema de comunicaciones para la red de sensores, es importante trabajar en las estrategias de detección de fallos, y restablecimiento de los enlaces de comunicaciones, que permitan auto configuración de la red, garantizando así la correcta transferencia de los datos en condiciones críticas de operación.

La adecuación de un Gateway en el nodo concentrador permitirá explorar el uso de diversas tecnologías de transmisión de datos de mayor alcance como por ejemplo enlaces de microondas, fibra óptica, Wi-Fi. Este esquema permitirá compartir información con otras microrredes obteniendo así un sistema más robusto.

La busqueda de nuevas fuentes de generación de energía ha llevado a que distintos países impulsen la construcción de microrredes. Estas microrredes cumplen los estándares de distribución correspondientes a la zona en donde se encuentren. Por ello se recomienda que se genere un prototipo que sea capaz de trabajar en cualquier microrred alrededor del mundo.

Con la información recolectada por la red de nodos sensores es posible determinar el comportamiento de la microrred. Por ende se sugiere que el software diseñado se conecte con una herramienta de procesamiento de recursos que permita gestionar los distintos elementos generadores de la microrred para optimizar el uso de la energía, conformando de esta manera todo un sistema AMI.

Se sugiere que el nodo sensor sea capaz de determinar la dirección del flujo de corriente de la red eléctrica.

10 REFERENCIAS.

[1] E. Pervasive and W. Sensor, “Slides - IEEE Std. 802.15.4,” *Group*, 2005.

[2] H. Oudani, “Comparative Study and Simulation of Flat and Hierarchical Routing Protocols for Wireless Sensor Network,” 2016.

[3] S. Krit, M. Benaddy, and B. El Habil, “Reliability of Transport Data And Energy Efficient in Wireless Sensor Networks : A Literature Survey,” pp. 1–4, 2016.

[4] A. D. Wood and J. A. Stankovic, “Denial of service in sensor networks,” *Computer (Long. Beach. Calif).*, vol. 35, no. 10, pp. 54–62, 2002.

[5] M. Kabrane, “Urban Mobility in Smart Cities Using Low-Cost and Energy-Saving Wireless Sensor Networks,” 2016.

[6] P. Nsonga and T. S. Ustun, “Integration of Communication Standards in Electrical Vehicle Ad-Hoc Networks for Smartgrid Support,” *IEEE Int. Conf. Emerg. Technol. Innov. Bus. Pract. Transform. Soc.*, 2016.

[7] M. A. Rahman, P. Bera, and E. Al-shaer, “SmartAnalyzer : A Noninvasive Security Threat Analyzer for AMI Smart Grid,” pp. 2255–2263, 2012.

[8] A. Payal, C. S. Rai, and B. V. R. Reddy, “Experimental Analysis of Some Radio Propagation Models for Smart Wireless Sensor Networks Applications,” pp. 338–342, 2015.

[9] N. J. J. and I. K. Y. S. H. Choi, S. J. Kang, “The design of outage management system utilizing meter information based on AMI (Advanced Metering Infrastructure) system,” pp. 1–7, 2011.

[10] I. Joe, J. Y. Jeong, and F. Zhang, “Design and Implementation of AMI System using Binary CDMA for Smart Grid,” pp. 544–549, 2013.

[11] E. Hossain, E. Kabalci, R. Bayindir, and R. Perez, “Microgrid testbeds around the world: State of art,” *Energy Convers. Manag.*, vol. 86, pp. 132–153, 2014.

[12] A. B. Ahmed, L. Weber, and A. Nasiri, “Microgrid Communications : State of the Art and Future Trends,” pp. 780–785, 2014.

[13] M. Manigandan and B. Basavaraja, “Active and reactive power control of MICROGRID using wireless technology (ZigBee 2.4 GHz),” *Int. Conf. Sustain. Energy Intell. Syst. (SEISCON 2011)*, no. Seiscon, pp. 564–570, 2011.

[14] G. Pepermans, J. Driesen, D. Haeseldonckx, R. Belmans, and W. D’haeseleer, “Distributed generation: Definition, benefits and issues,” *Energy Policy*, vol. 33, no. 6, pp. 787–798, 2005.

[15] H. B. Püttgen, P. R. Macgregor, and F. C. Lambert, “Distributed Generation: Semantic Hype or the Dawn of a New Era?,” *IEEE Power Energy Mag.*, vol. 1, no. 1, pp. 22–29, 2003.

[16] J. M. Guerrero, M. Chandorkar, T. Lee, and P. C. Loh, “Advanced Control Architectures for Intelligent Microgrids; Part I: Decentralized and Hierarchical Control,” *Ind. Electron. IEEE Trans.*, vol. 60, no. 4, pp. 1254–1262, 2013.

[17] R. Velázquez, H. G. Sarmiento, J. L. Silva, G. Vidrio, and R. Nieva, “Introducción al concepto de microrredes,” *Bol. IIE, Tendencias tecnológicas*, no. figura 1, pp. 102–108, 2010.

[18] J. M. Guerrero, J. C. Vásquez, J. Matas, M. Castilla, and L. García de Vicuna, “Control strategy for flexible microgrid based on parallel line-interactive UPS systems,” *IEEE Trans. Ind. Electron.*, vol. 56, no. 3, pp. 726–736, 2009.

[19] R. Burrett, C. Clini, and R. Dixon, “Renewables Global Status Report 2009 Update,” *Update*, pp. 1–31, 2009.

[20] D. Kondoleon, L. Ten-Hope, T. Surles, and R. L. Therkelsen, “The CERTS MicroGrid Concept,” *Integr. Distrib. Energy Resour. – CERTS MicroGrid Concept*, no. October, p. 32, 2003.

[21] O. G. Eduardo Prieto, Gerard Clariana, Andreu Vidal, Guillem Vinyals, Roberto Villafáfila, “Desarrollo de una microredde laboratorio para estudios de integración de energías renovables en la red eléctrica,” *II Congr. Smart Grids*.

[22] C. K. M. N. O. S. Alexander, “Fundamentos de Circuitos Electricos.” Mc Graw Hill, México. D.F, 2006.

[23] Daniel W. Hart, “ELECTRONICA DE POTENCIA.” Madrid, 2001.

[24] P. H. Corredor, P. E. Gomez C, R. Présiga Duque, L. J. Zuluaga L., and J. E. Tobón B., “Gestión de la calidad de la frecuencia en sistemas de distribución,” pp. 1–6.

[25] M. Barnes, J. Kondoh, H. Asano, J. Oyarzabal, G. Ventakaramanan, R. Lasseter, N. Hatziargyriou, and T. Green, “Real-World MicroGrids-An Overview,” *2007 IEEE Int. Conf. Syst. Syst. Eng.*, pp. 1–8, 2007.

[26] M. Gonzáles, “Simulación de un algoritmo de enrutamiento para redes de sensores inalámbricos,” pp. 1–79, 2012.

[27] J. L. Hill, “System Architecture for Wireless Sensor Networks,” University of Californica, Berkeley, 2003.

[28] A. M. V. Erazo, S. Y. Tang, and Y. Qian, “Wireless Sensor Network Communication Architecture for Wide-Area Large Scale Soil Moisture Estimation and Wetlands Monitoring,” no. 424546, p. 24, 2006.

[29] S. T. Babiker and A. F. M. Bashir, “Security of Wireless Sensor Networks for Monitoring System,” vol. 2, no. 2, pp. 5–9, 2015.

[30] H. Farhangi, “The path of the smart grid,” *IEEE Power Energy Mag.*, vol. 8, no. 1, pp. 18–28, 2010.

[31] C. Andrés, D. Andrade, and J. C. Hernández, “Smart Grid: Las TICs y la modernización de las redes de energía eléctrica-Estado del Arte,” *Sist. y …*, vol. 9, pp. 53–81, 2011.

[32] E. Pervasive and W. Sensor, “IEEE Std. 802.15.4,” *Group*, 2005.

[33] E. R. Ramos and A. S. Rodríguez, “Operación y control de microredes.”

[34] I. T. Monroy, N. G. González, A. Caballero, K. Prince, D. Zibar, T. Gibbon, X. Yu, and J. B. Jensen, “Convergencia de sistemas de comunicación ópticos e inalámbricos,” *Opt. Pura y Apl.*, vol. 42, no. 2, pp. 83–90, 2009.

[35] S. Barbarossa, S. Sardellitti, and P. Di Lorenzo, “Communicating While Computing: Distributed mobile cloud computing over 5G heterogeneous networks,” *IEEE Signal Process. Mag.*, vol. 31, no. 6, pp. 45–55, 2014.

[36] F. Bernal, N. Cancino, and E. Gaona, “Measurement infrastructure on microgrids using wireless sensors,” *2015 Work. Eng. Appl. - Int. Congr. Eng. WEA 2015*, 2015.

[37] E. Gunther and A. Snyder, “Smart grid standards assessment and recommendations for adoption and development,” *Prep. Calif. …*, no. October, pp. 1–33, 2009.

[38] E. G. Herbert Rojas Cubides, César Rodríguez Trujillo, “Redes de Ingeniería Redes de Ingeniería Redes de Ingeniería,” vol. 4, no. 2, pp. 37–44, 2015.

[39] J. I. Gonzalez and J. G. Fernández, “Desarrollo e implementación de una red inalámbrica de sensores de temperatura y humedad,” Universidad Publica de Navarra, 2014.

[40] R. E. D. Inalámbrica, D. E. S. Ambientales, M. El, U. S. O. D. E. La, and T. Zigbee, “Red inalambrica de sensores ambientales mediante el uso de la tecnologia zigbee,”

2013.

[41] A. R. MADELEN and G. V. J. ALBERTO, “IMPLEMENTACIÓN DE UN PROTOTIPO PARA AUTOMATIZACION DE RIEGOS.,” *Univ. CATÓLICA Pop. DEL RISARALDA*, 2009.

[42] M. V. Coronel Gutiérrez, “Estudio para la implementación del sistema de infraestructura de medición avanzada (AMI) en la empresa eléctrica regional centro sur CA,” 2011.

[43] P. L. Sub-committee, “The Role of Load Research in Automated Meter Infrastructure / Meter Data Management Initiatives,” no. September, 2008.

[44] A. . Fallis, “No Title No Title,” *J. Chem. Inf. Model.*, vol. 53, no. 9, pp. 1689–1699, 2013.

[45] E. Power, “Advanced Metering Infrastructure (AMI),” 2007.

[46] W. Callender, W. Baker, B. Laningham, E. Rogers, M. Williamson, and D. Hayes, “The Role of Load Research in Automated Meter Infrastructure / Meter Data Management Initiatives,” no. September, p. 15, 2008.

[47] S. W. Luan, J. H. Teng, L. C. Hwang, and C. S. Chen, “Wireless network deployment for intelligent fault indicators - A case study with ZigBee,” *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON*, pp. 1246–1250, 2011.

[48] UtilitiPoint-International, “The Department of Energy Raises Bar for Smart Grid / Smart Metering Initiatives – Key Strategies for Success,” pp. 1–13, 2010.

[49] M. Alvarado, “Servicios de Medición Avanzada (AMI) para Redes Inteligentes y su Adaptabilidad en el Marco de la Legislación Ecuatoriana,” p. 159, 2010.

[50] CODENSA S.A. ESP, “NIVELES DE TENSIÓN DE CONEXIÓN DE CARGAS DE CLIENTES | Likinormas.” [Online]. Available: http://likinormas.micodensa.com/Norma/otros/niveles_tension_conexion_cargas_clientes. [Accessed: 03-Nov-2016].

[51] LA COMISIÓN DE REGULACIÓN DE ENERGÍA Y GAS, “RESOLUCIÓN No.097,” 2008. [Online]. Available: http://apolo.creg.gov.co/Publicac.nsf/Indice01/Resolucion-2008-Creg097-2008. [Accessed: 03-Nov-2016].

[52] J. Dignani, “Análisis del protocolo zigbee,” p. 42, 2011.

[53] Texas Intruments, “Code Composer Studio (CCS) Integrated Development Environment (IDE) - CCSTUDIO - TI Tool Folder,” 2016. [Online]. Available:

http://www.ti.com/tool/ccstudio. [Accessed: 15-Sep-2016].

[54] T. Intruments, “controlSUITE™ Software Suite: Essential Software and Development Tools for C2000™ Microcontrollers - CONTROLSUITE - TI Software Folder.” [Online]. Available: http://www.ti.com/tool/controlsuite.

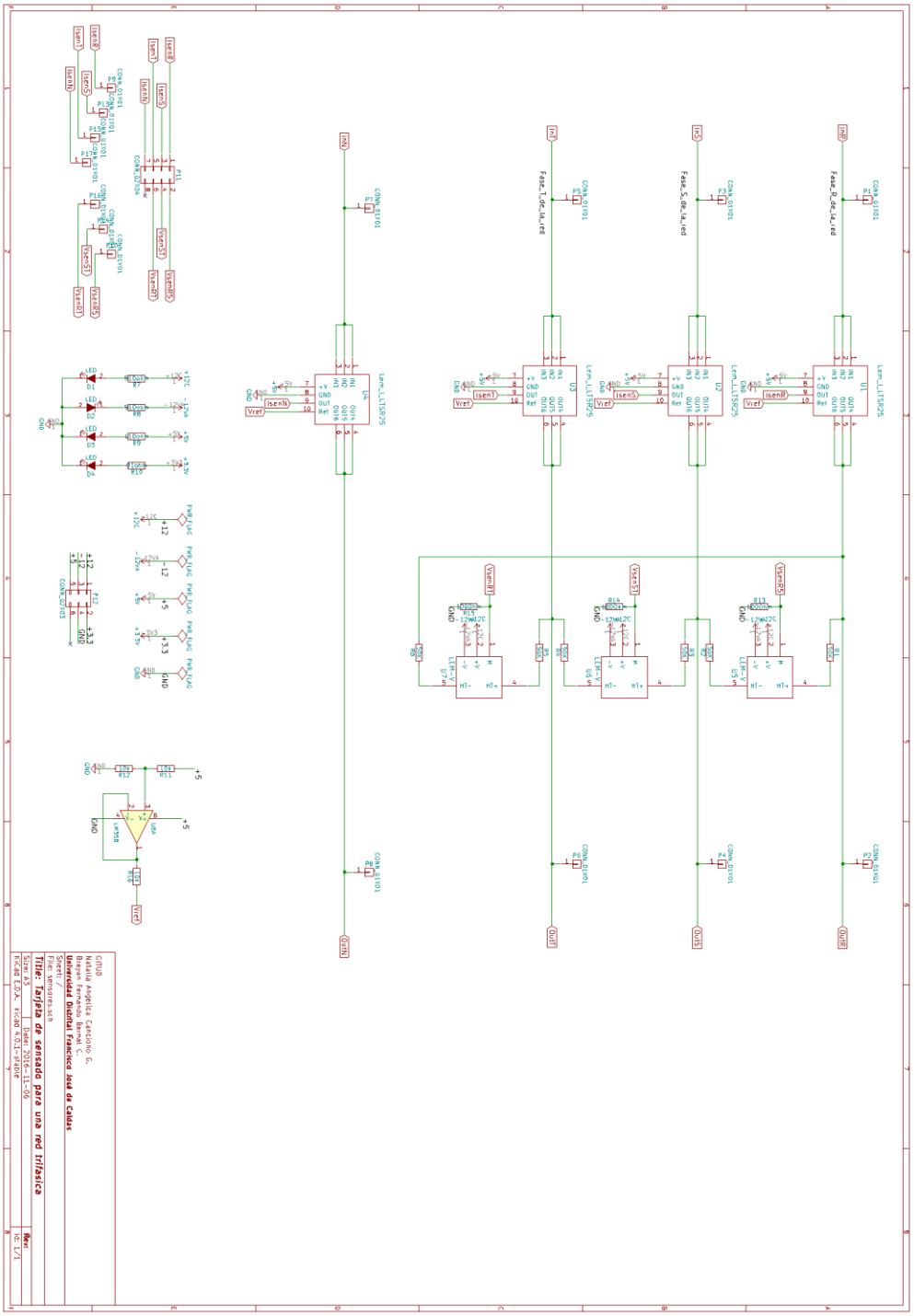
[55] NetBeans, “Welcome to NetBeans.” [Online]. Available: https://netbeans.org/. [Accessed: 20-Sep-2016].

[56] PosgreSQL, “PostgreSQL: The world’s most advanced open source database.” [Online]. Available: https://www.postgresql.org/. [Accessed: 15-Aug-2016].

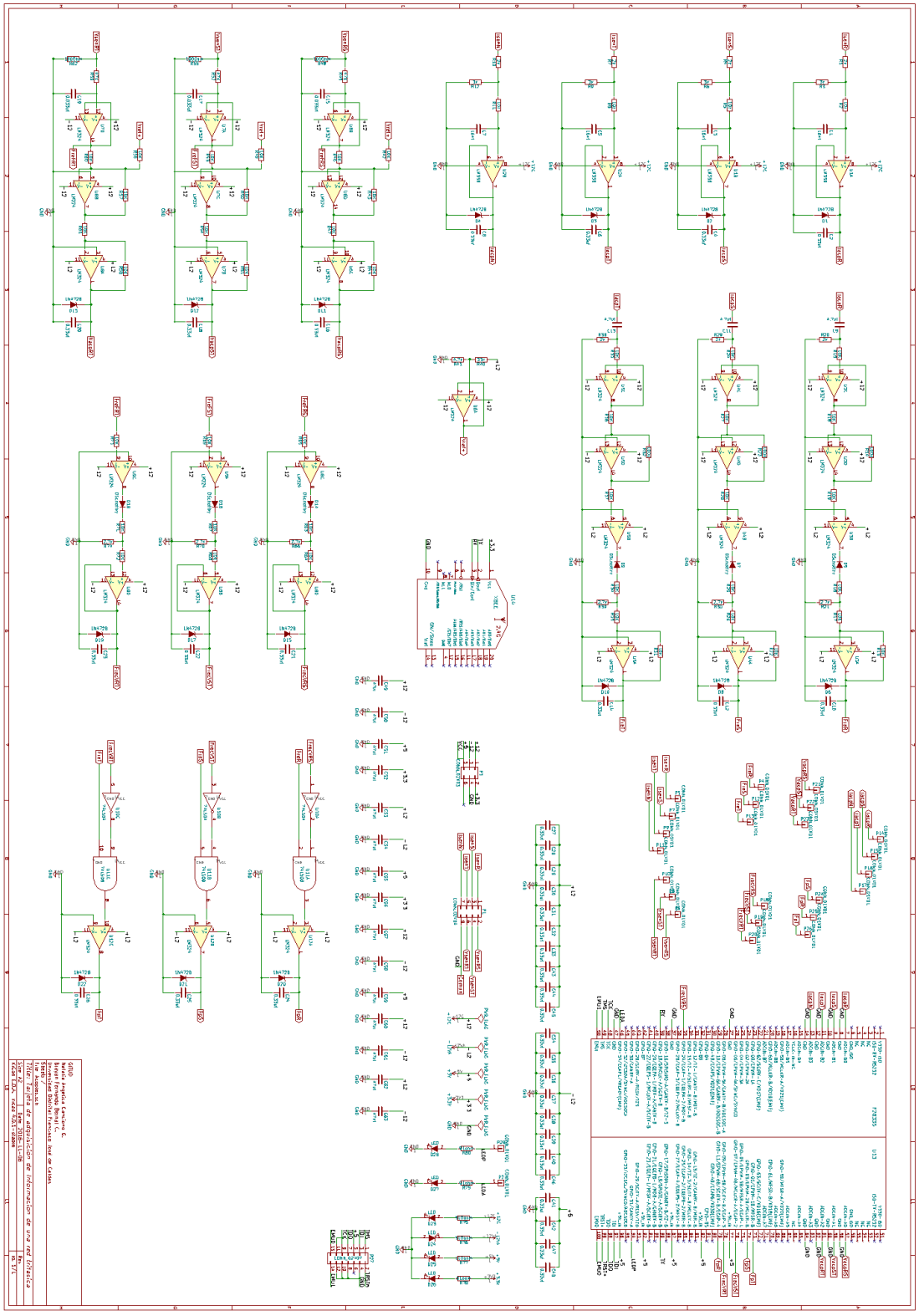
[57] DIGI, “XCTU Software Product Detail - Digi International.” [Online]. Available: https://www.digi.com/support/productdetail?pid=3352. [Accessed: 16-Jul-2016].

11
ANEXOS.

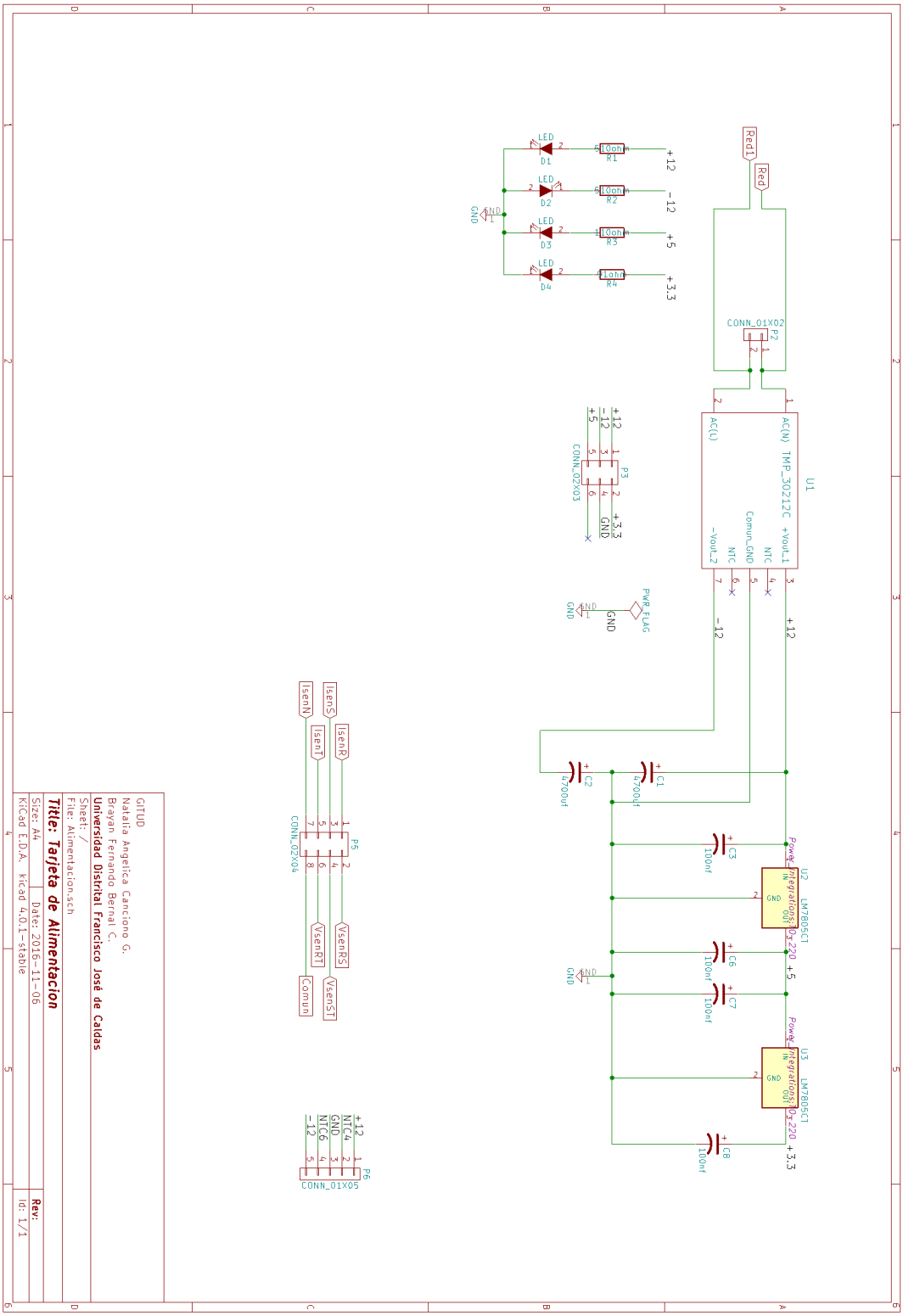
11.1 DIAGRAMA ELÉCTRICO DE ESQUEMA DE SENSADO.



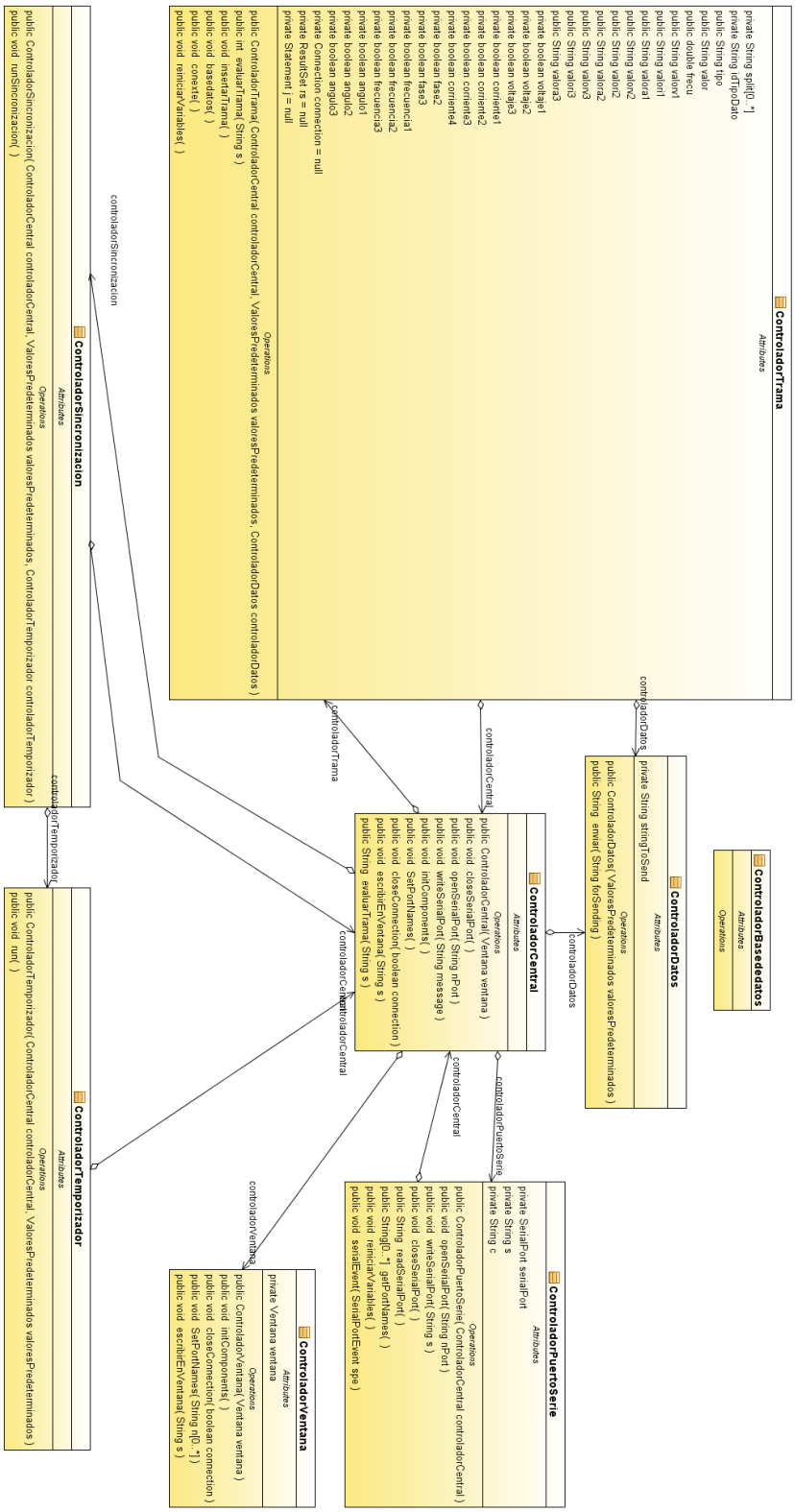
11.2DIAGRAMA ELÉCTRICO DEL ACOPLE DE SEÑALES, TARJETA DE PROCESAMIENTO Y COMUNICACIÓN DEL NODO SENSOR.



11.3DIAGRAMA ELÉCTRICO DE LA TARJETA DE ALIMENTACIÓN.



11.4 DIAGRAMA DE FLUJO DEL SOFTWARE GENERADO.



11.5CÓDIGO PRUEBA PARA CALIBRACIÓN.

```
#include "DSP28x_Project.h"     // Device Headerfile and Examples Include File
#include <string.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

////VARIABLES RAM /////
extern Uint16 RamfuncsLoadStart;
extern Uint16 RamfuncsLoadEnd;
extern Uint16 RamfuncsRunStart;

////////CONFIGURACION////////
void ADCconfiruracion(void);

//////CICLO////////
void ADCLoop(void);

void loop2(void);
//////VARIABLES CLK ADC///////
#define ADC_MODCLK 0x3 // HSPCLK = SYSCLKOUT/2*ADC_MODCLK2 = 150/(2*3)   = 25.0
MHz
#define ADC_CKPS   0x9   // ADC module clock = HSPCLK/2*ADC_CKPS   =
25.0MHz/(2*9) = 1.38MHz
#define ADC_SHCLK  0xf   // S/H VENTANA DEL PERIODO DEL MODULO ADC           =
16 ADC clocks

///FACTOR MULTIPLICATIVO CONVERSION 0-3V///
double multiplica=0.0007326007326007326;

//VECTORES
   //// VOLTAJES///
double MVAB[185],MVBC[185],MVCA[185];

  ////CORRIENTES///
double MIA[185],MIB[185],MIC[185],MIN[185];

   ///MAXIMOS CALIBRAR///
double VAB=0,VBC=0,VCA=0;
double IA=0,IB=0,IC=0,IN=0;


void main(void)
{
   InitSysCtrl();

        EALLOW;
        SysCtrlRegs.HISPCP.all = ADC_MODCLK;    // HSPCLK = SYSCLKOUT/ADC_MODCLK
        EDIS;
        DINT;

   InitPieCtrl();

   IER = 0x0000;
   IFR = 0x0000;
```

```
   InitPieVectTable();
   MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd, &RamfuncsRunStart);

   InitFlash();
   InitAdc();

         EINT;   // Enable Global interrupt INTM
         ERTM;   // Enable Global realtime interrupt DBGM

         ADCconfiruracion();
for(;;)
{


    ADCLoop();
    loop2();
    VAB=0;  VBC=0;  VCA=0;
    IA=0;  IB=0;  IC=0; IN=0;

}
}


////////////////////////////////////////////////////////////////////////
///////CONFIGURACIONES//////////////
void ADCconfiruracion(void)
{

    AdcRegs.ADCTRL1.bit.ACQ_PS = ADC_SHCLK;  //
    AdcRegs.ADCTRL3.bit.ADCCLKPS = ADC_CKPS; //

    AdcRegs.ADCTRL2.bit.SOC_SEQ1 = 1;        //
    AdcRegs.ADCTRL3.bit.SMODE_SEL = 1;       //
    AdcRegs.ADCTRL1.bit.SEQ_CASC = 1;        //

    AdcRegs.ADCTRL2.bit.INT_ENA_SEQ1= 1;     //
    AdcRegs.ADCTRL2.bit.RST_SEQ1 = 1;        //

    AdcRegs.ADCCHSELSEQ1.bit.CONV00 = 0x0;   //
    AdcRegs.ADCCHSELSEQ1.bit.CONV01 = 0x1;   //
    AdcRegs.ADCCHSELSEQ1.bit.CONV02 = 0x2;   //
    AdcRegs.ADCCHSELSEQ1.bit.CONV03 = 0x3;   //
    AdcRegs.ADCCHSELSEQ2.bit.CONV04 = 0x4;   //
    AdcRegs.ADCCHSELSEQ2.bit.CONV05 = 0x5;   //
    AdcRegs.ADCCHSELSEQ2.bit.CONV06 = 0x6;   //
    AdcRegs.ADCMAXCONV.bit.MAX_CONV1=0x0006; //

}
////////////////////////////////////////////////////////////////////////
                  ///////CICLOS//////////////

void ADCLoop(void)
{
```

```
int i;
 /// LIMPIEZA VARIABLES Y VECTORES//
   for (i=0; i<185; i++)
   {
     MVAB[i] = 0;  MVBC[i] = 0; MVCA[i] = 0;
     MIA[i] = 0;   MIB[i] = 0;  MIC[i] = 0 , MIN[i] = 0;

  //  VAB=0;  VBC=0;  VCA=0;
    //IA=0;  IB=0;  IC=0; IN=0;

    }
//// CAPTURA DE DATOS ////

    for (i=0; i<185; i++)
     {
          AdcRegs.ADCTRL2.bit.SOC_SEQ1 = 1;

          MVAB[i]=(AdcRegs.ADCRESULT0>>4)*multiplica;
          MVBC[i]=(AdcRegs.ADCRESULT2>>4)*multiplica;
          MVCA[i]=(AdcRegs.ADCRESULT4>>4)*multiplica;
          MIA[i]=(AdcRegs.ADCRESULT1>>4)*multiplica;
          MIB[i]=(AdcRegs.ADCRESULT3>>4)*multiplica;
          MIC[i]=(AdcRegs.ADCRESULT5>>4)*multiplica;
          MIN[i]=(AdcRegs.ADCRESULT7>>4)*multiplica;

         while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {} // Wait for interrupt
          AdcRegs.ADCST.bit.INT_SEQ1_CLR = 1;
     }

// MAXIMOS PARA CALIBRAR //
   for (i=1;i<185;i++)
  {
      if (MVAB[i] > MVAB[i-1] && MVAB[i]>VAB)
      {
          VAB=MVAB[i];
      }
    }

   for (i=1;i<185;i++)
  {
      if (MVBC[i]> MVBC[i-1] && MVBC[i]>VBC)
      {
          VBC=MVBC[i];
      }
    }
   for (i=1;i<185;i++)
  {
      if (MVCA[i]> MVCA[i-1] && MVCA[i]>VCA)
      {
          VCA=MVCA[i];
      }
    }
   for (i=1;i<185;i++)
  {
      if (MIA[i]> MIA[i-1] && MIA[i]>IA)
```

```
      {
          IA=MIA[i];
      }
    }

   for (i=1;i<185;i++)
    {
        if (MIB[i]> MIB[i-1] && MIB[i]>IB)
        {
            IB=MIB[i];
        }
      }
   for (i=1;i<185;i++)
    {
        if (MIC[i]> MIC[i-1] && MIC[i]>IC)
        {
            IC=MIC[i];
        }
      }
   for (i=1;i<185;i++)
     {
         if (MIN[i]> MIN[i-1] && MIN[i]>IN)
         {
             IN=MIN[i];
         }
       }

}
void loop2(void)
{long m;
    for (m=0;m<10000000;m++)
          {

          }
}
```

11.6CÓDIGO PRUEBA CÁLCULO DE VALORES RMS.

```
#include "DSP28x_Project.h"     // Device Headerfile and Examples Include File
#include <string.h>
#include <math.h>
#include <stdio.h>
#include <stdlib.h>

////VARIABLES RAM /////
extern Uint16 RamfuncsLoadStart;
extern Uint16 RamfuncsLoadEnd;
extern Uint16 RamfuncsRunStart;

////////CONFIGURACION////////
void ADCconfiruracion(void);

//////CICLO////////
void ADCLoop(void);
```

```
void loop2(void);
//////VARIABLES CLK ADC///////
#define ADC_MODCLK 0x3 // HSPCLK = SYSCLKOUT/2*ADC_MODCLK2 = 150/(2*3)   = 25.0
MHz
#define ADC_CKPS   0x9   // ADC module clock = HSPCLK/2*ADC_CKPS   =
25.0MHz/(2*9) = 1.38MHz
#define ADC_SHCLK  0xf   // S/H VENTANA DEL PERIODO DEL MODULO ADC           =
16 ADC clocks

///FACTOR MULTIPLICATIVO CONVERSION 0-3V///
double multiplica=0.0007326007326007326;

//VECTORES
   //// VOLTAJES///
double MVAB[185],MVBC[185],MVCA[185];

  ////CORRIENTES///
double MIA[185],MIB[185],MIC[185],MIN[185];

   //VOLTAJES PROMEDIOS//
double MVAB2[20],MVBC2[20],MVCA2[20];

///VARIABLES PARA CICLOS
int j=0;
int k=0;
int y;
// VALORES DC Y RMS
  ////VOLTAJES
double FVABDC,FVBCDC,FVCADC;
double FVABRMS,FVBCRMS,FVCARMS;
double FVABRMS2,FVBCRMS2,FVCARMS2; // raices
double VABRMS,VBCRMS,VCARMS; // raiz final para enviar
/// PROMEDIO DE RMS
double PVABRMS,PVBCRMS,PVCARMS;

  ////CORRIENTES
double FIADC,FIBDC,FICDC,FINDC;
double FIARMS,FIBRMS,FICRMS,FINRMS;
double FIARMS2,FIBRMS2,FICRMS2,FINRMS2;// raices
double IARMS,IBRMS,ICRMS,INRMS;// raiz final para enviar

int VVABRMS,VVBCRMS,VVCARMS;
int VIARMS,VIBRMS,VICRMS,VINRMS;


void main(void)
{
   InitSysCtrl();

        EALLOW;
        SysCtrlRegs.HISPCP.all = ADC_MODCLK;    // HSPCLK = SYSCLKOUT/ADC_MODCLK
        EDIS;
        DINT;

   InitPieCtrl();
```

```
   IER = 0x0000;
   IFR = 0x0000;

   InitPieVectTable();
   MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd, &RamfuncsRunStart);

   InitFlash();
   InitAdc();

         EINT;   // Enable Global interrupt INTM
         ERTM;   // Enable Global realtime interrupt DBGM

         ADCconfiruracion();
for(;;){
         for (k=0; k<20; k++)
{
    ADCLoop();

    VVABRMS= PVABRMS*100;   VVBCRMS=PVBCRMS*100;    VVCARMS=PVCARMS*100;
    VIARMS =IARMS*100;   VIBRMS=IBRMS*100;   VICRMS=ICRMS*100;
VINRMS=INRMS*100;
    }

    loop2();
    for (y=0; y<185; y++)
    {
        k=0; j=0;
      MVAB[y] = 0;  MVBC[y] = 0; MVCA[y] = 0;
      MIA[y] = 0;   MIB[y] = 0;  MIC[y] = 0 , MIN[y] = 0;

      FVABDC=0;   FVBCDC=0;   FVCADC=0;
      FVABRMS=0;  FVBCRMS=0;  FVCARMS=0;

      FIADC=0;    FIBDC=0;   FICDC=0; FINDC=0;
      FIARMS=0;   FIBRMS=0;  FICRMS=0; FINRMS=0;
      VVABRMS= 0;   VVBCRMS=0;    VVCARMS=0;
      VIARMS =0;   VIBRMS=0;   VICRMS=0;   VINRMS=0;
    }

}
}


////////////////////////////////////////////////////////////////////////////
////////CONFIGURACIONES//////////////
void ADCconfiruracion(void)
{

    AdcRegs.ADCTRL1.bit.ACQ_PS = ADC_SHCLK;  //
      AdcRegs.ADCTRL3.bit.ADCCLKPS = ADC_CKPS; //

      AdcRegs.ADCTRL2.bit.SOC_SEQ1 = 1;        //
      AdcRegs.ADCTRL3.bit.SMODE_SEL = 1;       //
      AdcRegs.ADCTRL1.bit.SEQ_CASC = 1;        //
```

```
      AdcRegs.ADCTRL2.bit.INT_ENA_SEQ1= 1;     //
      AdcRegs.ADCTRL2.bit.RST_SEQ1 = 1;        //

      AdcRegs.ADCCHSELSEQ1.bit.CONV00 = 0x0;   // A0-B0
      AdcRegs.ADCCHSELSEQ1.bit.CONV01 = 0x1;   // A1-B1
      AdcRegs.ADCCHSELSEQ1.bit.CONV02 = 0x2;   // A2-B2
      AdcRegs.ADCCHSELSEQ1.bit.CONV03 = 0x3;   // A3-B3
      AdcRegs.ADCCHSELSEQ1.bit.CONV03 = 0x3;   //
      AdcRegs.ADCCHSELSEQ2.bit.CONV04 = 0x4;   //
      AdcRegs.ADCCHSELSEQ2.bit.CONV05 = 0x5;   //


      AdcRegs.ADCMAXCONV.bit.MAX_CONV1=0x0006; //

}
////////////////////////////////////////////////////////////////////////////
////////CICLOS//////////////

void ADCLoop(void)
{

int i;
 /// LIMPIEZA VARIABLES Y VECTORES//
   for (i=0; i<185; i++)
   {
     MVAB[i] = 0;  MVBC[i] = 0; MVCA[i] = 0;
     MIA[i] = 0;   MIB[i] = 0;  MIC[i] = 0 , MIN[i] = 0;

     FVABDC=0;   FVBCDC=0;   FVCADC=0;
     FVABRMS=0;  FVBCRMS=0;  FVCARMS=0;

     FIADC=0;    FIBDC=0;   FICDC=0; FINDC=0;
     FIARMS=0;   FIBRMS=0;  FICRMS=0; FINRMS=0;
   }
//// CAPTURA DE DATOS ////

    for (i=0; i<185; i++)
     {
          AdcRegs.ADCTRL2.bit.SOC_SEQ1 = 1;

          MVAB[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT0>>4))*multiplica))-
1.653271)/0.003272)+50.1);
          MVBC[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT2>>4))*multiplica))-
1.599372)/0.003044)+49.8);
          MVCA[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT4>>4))*multiplica))-
1.726426)/0.003038)+49.7);

          MIA[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT1>>4))*multiplica))-
1.594349)/0.021908)+2.1);
          MIB[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT3>>4))*multiplica))-
1.585767)/0.021340)+2.14);
          MIC[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT5>>4))*multiplica))-
1.591941)/0.021034)+2.14);
```

```
          MIN[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT7>>4))*multiplica))-
1.513762)/0.019108));

         while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {} // Wait for interrupt
          AdcRegs.ADCST.bit.INT_SEQ1_CLR = 1;
     }


/// VALORES DC Y RMS ///

   for (i=1;i<185;i++)
  {
      FVABDC=(MVAB[i]/184)+FVABDC; FVBCDC=(MVBC[i]/184)+FVBCDC;
FVCADC=(MVCA[i]/184)+FVCADC;
      FIADC=(MIA[i]/184)+FIADC; FIBDC=(MIB[i]/184)+FIBDC;
FICDC=(MIC[i]/184)+FICDC;

      FVABRMS=(((MVAB[i]*MVAB[i])/184)+FVABRMS);
FVBCRMS=(((MVBC[i]*MVBC[i])/184)+FVBCRMS);
FVCARMS=(((MVCA[i]*MVCA[i])/184)+FVCARMS);
      FIARMS=(((MIA[i]*MIA[i])/184)+FIARMS);
FIBRMS=(((MIB[i]*MIB[i])/184)+FIBRMS);FICRMS=(((MIC[i]*MIC[i])/184)+FICRMS);FINR
MS=(((MIN[i]*MIN[i])/184)+FINRMS);
  }

   FVABRMS2=sqrt(FVABRMS);    FVBCRMS2=sqrt(FVBCRMS);    FVCARMS2=sqrt(FVCARMS);
//
   // RAIZ FINAL RMS-DC
   VABRMS=sqrt((FVABRMS2*FVABRMS2)-(FVABDC*FVABDC));
VBCRMS=sqrt((FVBCRMS2*FVBCRMS2)-(FVBCDC*FVBCDC));
VCARMS=sqrt((FVCARMS2*FVCARMS2)-(FVCADC*FVCADC));

   MVAB2[j]=VABRMS;  MVBC2[j]=VCARMS;  MVCA2[j]=VCARMS;

   FIARMS2=sqrt(FIARMS); FIBRMS2=sqrt(FIBRMS); FICRMS2=sqrt(FICRMS);
FINRMS2=sqrt(FINRMS);

   IARMS=sqrt((FIARMS2*FIARMS2)-(FIADC-FIADC)); IBRMS=sqrt((FIBRMS2*FIBRMS2)-
(FIBDC-FIBDC)); ICRMS=sqrt((FICRMS2*FICRMS2)-(FICDC-FICDC));
INRMS=sqrt((FINRMS2*FINRMS2)-(FIADC-FINDC));

   j++;
// promedio VOLTAJES //
     if (j==20)
    {
        PVABRMS=( MVAB2[0]+ MVAB2[1]+ MVAB2[2]+ MVAB2[3] + MVAB2[4]+ MVAB2[5]+
MVAB2[6]+ MVAB2[7]+ MVAB2[8]+ MVAB2[9]+
                MVAB2[10]+ MVAB2[11]+ MVAB2[12]+ MVAB2[13] + MVAB2[14]+
MVAB2[15]+ MVAB2[16]+ MVAB2[17]+ MVAB2[18]+ MVAB2[19])/20;
        PVBCRMS=( MVBC2[0]+ MVBC2[1]+ MVBC2[2]+ MVBC2[3] + MVBC2[4]+ MVBC2[5]+
MVBC2[6]+ MVBC2[7]+ MVBC2[8]+ MVBC2[9]+
                        MVBC2[10]+ MVBC2[11]+ MVBC2[12]+ MVBC2[13] + MVBC2[14]+
MVBC2[15]+ MVBC2[16]+ MVBC2[17]+ MVBC2[18]+ MVBC2[19])/20;
        PVCARMS=( MVCA2[0]+ MVCA2[1]+ MVCA2[2]+ MVCA2[3] + MVCA2[4]+ MVCA2[5]+
MVCA2[6]+ MVCA2[7]+ MVCA2[8]+ MVCA2[9]+
```

```
                                MVCA2[10]+ MVCA2[11]+ MVCA2[12]+ MVCA2[13] +
MVCA2[14]+ MVCA2[15]+ MVCA2[16]+ MVCA2[17]+ MVCA2[18]+ MVCA2[19])/20;

    }


 }
void loop2(void)
{long m;
    for (m=0;m<10000000;m++)
          {

          }
}
```

11.7 CÓDIGO PRUEBA VALORES DE FRECUENCIA, FASE Y ÁNGULO DE DESFASE.

```
#include "DSP28x_Project.h"     // Device Headerfile and Examples Include File

////INTERRUPCIONES////
__interrupt void cpu_timer0_isr(void);
__interrupt void cpu_timer1_isr(void);

__interrupt void ecap1_isr(void);
__interrupt void ecap2_isr(void);
__interrupt void ecap3_isr(void);

__interrupt void ecap4_isr(void);
__interrupt void ecap5_isr(void);
__interrupt void ecap6_isr(void);

////CONFIGURACION///
void InitECapture(void);
////CICLOS//
void Frecuencia1();
void Frecuencia2();
void Frecuencia3();
void Frecuencia4();
void Frecuencia5();
void Frecuencia6();

///VARIABLES ///

// variable activacion y desactivacion fases//
int true=0;
// timer//
Uint16 seg=0;
Uint16 min=0;

   /// frecuencias//
double Frec=0;
double Frec2=0;
double Frec3=0;
double Frec4=0;
```

```
double Frec5=0;
double Frec6=0;
 /// contafores para fases//
double contador=0;
double contador1=0;
double contador2=0;
 /// fases//
double fase2=0;
double fase3=0;
 /// angulos //
double desfase14=0;
double desfase24=0;
double desfase15=0;
double desfase25=0;
double desfase16=0;
double desfase26=0;


void main(void)
{
   InitSysCtrl();
// frecuencias  voltajes//
   InitECap1Gpio();
   InitECap2Gpio();
   InitECap3Gpio();
// angulos ///
   InitECap4Gpio();
   InitECap5Gpio();
   InitECap6Gpio();

   DINT;

   InitPieCtrl();


   IER = 0x0000;
   IFR = 0x0000;


   InitPieVectTable();


   EALLOW;  // This is needed to write to EALLOW protected registers
   PieVectTable.ECAP1_INT = &ecap1_isr;
   PieVectTable.ECAP2_INT = &ecap2_isr;
   PieVectTable.ECAP3_INT = &ecap3_isr;
   PieVectTable.ECAP4_INT = &ecap4_isr;
   PieVectTable.ECAP5_INT = &ecap5_isr;
   PieVectTable.ECAP6_INT = &ecap6_isr;
   PieVectTable.TINT0 = &cpu_timer0_isr;
   PieVectTable.XINT13 = &cpu_timer1_isr;

   EDIS;    // This is needed to disable write to EALLOW protected registers

   InitCpuTimers();   // For this example, only initialize the Cpu Timers
```

```
     ConfigCpuTimer(&CpuTimer0, 150,1000000);
     ConfigCpuTimer(&CpuTimer1, 150,1);


       CpuTimer0Regs.TCR.all = 0x4000; // Use write-only instruction to set TSS
bit = 0
       CpuTimer1Regs.TCR.all = 0x4000; // Use write-only instruction to set TSS
bit = 0


   InitECapture();


   IER |= M_INT4;  ///ECAP
   IER |= M_INT1;  //TIMER0
   IER |= M_INT13; //TIMER1

   CpuTimer1Regs.TCR.bit.TIE = 0;

   PieCtrlRegs.PIEIER1.bit.INTx7 = 1; //reloj por corregir

   EINT;
   ERTM;

}
////////////////////////////////////////////////////////////////////////////
////////INTERRUPCIONES//////////////
__interrupt void cpu_timer0_isr(void)
{
   seg=CpuTimer0.InterruptCount++;

   if (seg==3)
   {
       CpuTimer0.InterruptCount=0;
       min++;

   }
if (min==2)
{min=0;
    }
if(min==1 && seg==0)
{
      PieCtrlRegs.PIEIER4.bit.INTx1 = 1;

      PieCtrlRegs.PIEIER4.bit.INTx6 = 1;

      PieCtrlRegs.PIEIER4.bit.INTx4 = 1;
      PieCtrlRegs.PIEIER4.bit.INTx5 = 1;

}


   PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;
```

```
}

__interrupt void cpu_timer1_isr(void)
{
  contador= CpuTimer1.InterruptCount++;
   EDIS;
}


__interrupt void ecap1_isr(void)
{
   true++;

    if (true<3)
    {
   CpuTimer1Regs.TCR.bit.TIE = 1;
   PieCtrlRegs.PIEIER4.bit.INTx2 = 1;
        contador1=contador;
    }

    if (true>3 && true<6)
       {
      CpuTimer1Regs.TCR.bit.TIE = 0;
      PieCtrlRegs.PIEIER4.bit.INTx2 = 0;
      CpuTimer1.InterruptCount=0;
        }
    if (true>6 && true<9)
    {
        CpuTimer1Regs.TCR.bit.TIE = 1;
        PieCtrlRegs.PIEIER4.bit.INTx3 = 1;
        contador2=contador;
    }
    if (true>9 && true<11)
        {
       CpuTimer1Regs.TCR.bit.TIE = 0;
       PieCtrlRegs.PIEIER4.bit.INTx3 = 0;
       CpuTimer1.InterruptCount=0;
         }

    if (true>12)
            {

           PieCtrlRegs.PIEIER4.bit.INTx1 = 0;
           PieCtrlRegs.PIEIER4.bit.INTx2 = 0;
           PieCtrlRegs.PIEIER4.bit.INTx3 = 0;
           PieCtrlRegs.PIEIER4.bit.INTx4 = 1;
           PieCtrlRegs.PIEIER4.bit.INTx5 = 1;
           PieCtrlRegs.PIEIER4.bit.INTx6 = 1;
           CpuTimer1Regs.TCR.bit.TIE = 0;
           CpuTimer1.InterruptCount=0;
           true=0;

             }
```

```
       Frecuencia1();
   ECap1Regs.ECCLR.bit.CEVT4 = 1;
   ECap1Regs.ECCLR.bit.INT = 1;
   ECap1Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;
   fase2=((contador1-1006)/(28.91666))+120;
   fase3=((contador2-1006)/(28.91666))+120;
}

__interrupt void ecap2_isr(void)
{
    CpuTimer1.InterruptCount=0;
    Frecuencia2();
   ECap2Regs.ECCLR.bit.CEVT4 = 1;
   ECap2Regs.ECCLR.bit.INT = 1;
   ECap2Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;

}

__interrupt void ecap3_isr(void)
{
    CpuTimer1.InterruptCount=0;
    Frecuencia3();
   ECap3Regs.ECCLR.bit.CEVT4 = 1;
   ECap3Regs.ECCLR.bit.INT = 1;
   ECap3Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;

 }


__interrupt void ecap4_isr(void)
{
    Frecuencia4();
   ECap4Regs.ECCLR.bit.CEVT4 = 1;
   ECap4Regs.ECCLR.bit.INT = 1;
   ECap4Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;
}
__interrupt void ecap5_isr(void)
{

    Frecuencia5();
   ECap5Regs.ECCLR.bit.CEVT4 = 1;
   ECap5Regs.ECCLR.bit.INT = 1;
```

```
   ECap5Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;
}

__interrupt void ecap6_isr(void)
{

    Frecuencia6();
   ECap6Regs.ECCLR.bit.CEVT4 = 1;
   ECap6Regs.ECCLR.bit.INT = 1;
   ECap6Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;
}
////////////////////////////////////////////////////////////////////////////
////////CONFIGURACIONES//////////////


void InitECapture()
{
    //ECAP1   configura fases y frecuencias

    //ECAP4
        ECap1Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
        ECap1Regs.ECCLR.all = 0xFFFF;              // Clear all CAP interrupt
flags
        ECap1Regs.ECCTL1.bit.CAPLDEN = 0;          // Disable CAP1-CAP4 register
loads
        ECap1Regs.ECCTL2.bit.TSCTRSTOP = 0;        // Make sure the counter is
stopped
        // Configure peripheral registers
        ECap1Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
        ECap1Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
        ECap1Regs.ECCTL1.bit.CAP1POL = 0;          // Rising edge
        ECap1Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
        ECap1Regs.ECCTL1.bit.CAP3POL = 0;          // Rising edge
        ECap1Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
        ECap1Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
        ECap1Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
        ECap1Regs.ECCTL1.bit.CTRRST3 = 0;          // Difference operation
        ECap1Regs.ECCTL1.bit.CTRRST4 = 0;          // Difference operation
        ECap1Regs.ECCTL2.bit.SYNCI_EN = 1;         // Enable sync in
        ECap1Regs.ECCTL2.bit.SYNCO_SEL = 0;        // Pass through
        ECap1Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable capture units


        ECap1Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
        ECap1Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
        ECap1Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4 register
loads
        ECap1Regs.ECEINT.bit.CEVT4 = 1;            // 4 events = interrupt
```

```
       // cap5
        ECap2Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
        ECap2Regs.ECCLR.all = 0xFFFF;              // Clear all CAP interrupt
flags
        ECap2Regs.ECCTL1.bit.CAPLDEN = 0;          // Disable CAP1-CAP4 register
loads
        ECap2Regs.ECCTL2.bit.TSCTRSTOP = 0;        // Make sure the counter is
stopped
        // C2nfigure peripheral registers
        ECap2Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
        ECap2Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
        ECap2Regs.ECCTL1.bit.CAP1POL = 0;          // Falling edge
        ECap2Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
        ECap2Regs.ECCTL1.bit.CAP3POL = 0;          // Falling edge
        ECap2Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
        ECap2Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
        ECap2Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
        ECap2Regs.ECCTL1.bit.CTRRST3 = 0;          // Difference operation
        ECap2Regs.ECCTL1.bit.CTRRST4 = 0;          // Difference operation
        ECap2Regs.ECCTL2.bit.SYNCI_EN = 1;         // Enable sync in
        ECap2Regs.ECCTL2.bit.SYNCO_SEL = 0;        // Pass through
        ECap2Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable capture units


        ECap2Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
        ECap2Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
        ECap2Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4 register
loads
        ECap2Regs.ECEINT.bit.CEVT4 = 1;            // 4 events = interrupt

          //cap3
        ECap3Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
        ECap3Regs.ECCLR.all = 0xFFFF;              // Clear all CAP interrupt
flags
        ECap3Regs.ECCTL1.bit.CAPLDEN = 0;          // Disable CAP1-CAP4 register
loads
        ECap3Regs.ECCTL2.bit.TSCTRSTOP = 0;        // Make sure the counter is
stopped
         // Configure peripheral registers
        ECap3Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
        ECap3Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
        ECap3Regs.ECCTL1.bit.CAP1POL = 0;          // Falling edge
        ECap3Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
        ECap3Regs.ECCTL1.bit.CAP3POL = 0;          // Falling edge
        ECap3Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
        ECap3Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
        ECap3Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
        ECap3Regs.ECCTL1.bit.CTRRST3 = 0;          // Difference operation
        ECap3Regs.ECCTL1.bit.CTRRST4 = 0;          // Difference operation
        ECap3Regs.ECCTL2.bit.SYNCI_EN = 1;         // Enable sync in
        ECap3Regs.ECCTL2.bit.SYNCO_SEL = 0;        // Pass through
        ECap3Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable capture units
```

```
        ECap3Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
        ECap3Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
        ECap3Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4 register
loads
        ECap3Regs.ECEINT.bit.CEVT4 = 1;            // 4 events = interrupt
// 4 events = interrupt

//// configura para desfases y frecuencias
    //ECAP4

          ECap4Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
          ECap4Regs.ECCLR.all = 0xFFFF;              // Clear all CAP interrupt
flags
          ECap4Regs.ECCTL1.bit.CAPLDEN = 0;          // Disable CAP1-CAP4
register loads
          ECap4Regs.ECCTL2.bit.TSCTRSTOP = 0;        // Make sure the counter is
stopped
          // Configure peripheral registers
          ECap4Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
          ECap4Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
          ECap4Regs.ECCTL1.bit.CAP1POL = 1;          // Falling edge
          ECap4Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
          ECap4Regs.ECCTL1.bit.CAP3POL = 1;          // Falling edge
          ECap4Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
          ECap4Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
          ECap4Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
          ECap4Regs.ECCTL1.bit.CTRRST3 = 0;          // Difference operation
          ECap4Regs.ECCTL1.bit.CTRRST4 = 0;          // Difference operation
          ECap4Regs.ECCTL2.bit.SYNCI_EN = 1;         // Enable sync in
          ECap4Regs.ECCTL2.bit.SYNCO_SEL = 0;        // Pass through
          ECap4Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable capture units


          ECap4Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
          ECap4Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
          ECap4Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4
register loads
          ECap4Regs.ECEINT.bit.CEVT4 = 1;
   // cap5
      ECap5Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
      ECap5Regs.ECCLR.all = 0xFFFF;              // Clear all CAP interrupt
flags
      ECap5Regs.ECCTL1.bit.CAPLDEN = 0;          // Disable CAP1-CAP4 register
loads
      ECap5Regs.ECCTL2.bit.TSCTRSTOP = 0;        // Make sure the counter is
stopped
      // Configure peripheral registers
      ECap5Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
      ECap5Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
      ECap5Regs.ECCTL1.bit.CAP1POL = 1;          // Falling edge
      ECap5Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
      ECap5Regs.ECCTL1.bit.CAP3POL = 1;          // Falling edge
```

```
      ECap5Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
      ECap5Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
      ECap5Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
      ECap5Regs.ECCTL1.bit.CTRRST3 = 0;          // Difference operation
      ECap5Regs.ECCTL1.bit.CTRRST4 = 0;          // Difference operation
      ECap5Regs.ECCTL2.bit.SYNCI_EN = 1;         // Enable sync in
      ECap5Regs.ECCTL2.bit.SYNCO_SEL = 0;        // Pass through
      ECap5Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable capture units


      ECap5Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
      ECap5Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
      ECap5Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4 register
loads
      ECap5Regs.ECEINT.bit.CEVT4 = 1;            // 4 events = interrupt

      //cap6
      ECap6Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
      ECap6Regs.ECCLR.all = 0xFFFF;              // Clear all CAP interrupt
flags
      ECap6Regs.ECCTL1.bit.CAPLDEN = 0;          // Disable CAP1-CAP4 register
loads
      ECap6Regs.ECCTL2.bit.TSCTRSTOP = 0;        // Make sure the counter is
stopped
      // Configure peripheral registers
      ECap6Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
      ECap6Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
      ECap6Regs.ECCTL1.bit.CAP1POL = 1;          // Falling edge
      ECap6Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
      ECap6Regs.ECCTL1.bit.CAP3POL = 1;          // Falling edge
      ECap6Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
      ECap6Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
      ECap6Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
      ECap6Regs.ECCTL1.bit.CTRRST3 = 0;          // Difference operation
      ECap6Regs.ECCTL1.bit.CTRRST4 = 0;          // Difference operation
      ECap6Regs.ECCTL2.bit.SYNCI_EN = 1;         // Enable sync in
      ECap6Regs.ECCTL2.bit.SYNCO_SEL = 0;        // Pass through
      ECap6Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable capture units


      ECap6Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
      ECap6Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
      ECap6Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4 register
loads
      ECap6Regs.ECEINT.bit.CEVT4 = 1;            // 4 events = interrupt
}

////////////////////////////////////////////////////////////////////////////
////////CICLOS//////////////

void Frecuencia1()
{
    Uint32 P1;
    Uint32 P2;
```

```
    P1 = ECap1Regs.CAP2-ECap1Regs.CAP1; // Calculate 1st period
    P2 = ECap1Regs.CAP3-ECap1Regs.CAP2; // Calculate 2st period

        if(P1<P2)
        {
           Frec = 150e6/P1;
        }
        else if (P2<P1)
        {
           Frec = 150e6/P2;
        }

        ECap1Regs.TSCTR=0;

}

void Frecuencia2()
{
    Uint32 P12;
    Uint32 P22;

    P12 = ECap2Regs.CAP2-ECap2Regs.CAP1; // Calculate 1st period
    P22 = ECap2Regs.CAP3-ECap2Regs.CAP2; // Calculate 2st period


    if(P12<P22)
    {
       Frec2 = 150e6/P12;
    }
    else if (P22<P12)
    {
       Frec2 = 150e6/P22;
    }

    ECap2Regs.TSCTR=0;
}

void Frecuencia3()
{
    Uint32 P13;
    Uint32 P23;

    P13 = ECap3Regs.CAP2-ECap3Regs.CAP1; // Calculate 1st period
    P23 = ECap3Regs.CAP3-ECap3Regs.CAP2; // Calculate 2st period

    if(P13<P23)
    {
       Frec3 = 150e6/P13;
    }
    else if (P23<P13)
    {
       Frec3 = 150e6/P23;
    }
    ECap3Regs.TSCTR=0;
```

```
}
//// desfases
void Frecuencia4()
{
    Uint32 P14;
    Uint32 duty14;
    Uint32 duty24;

     P14 =  ECap4Regs.CAP3-ECap4Regs.CAP1; // Calculate 1st period
     Frec4 = 150e6/P14;

     duty14=ECap4Regs.CAP2-ECap4Regs.CAP1;
     duty24= ECap4Regs.CAP3-ECap4Regs.CAP2;
     desfase14=(duty14*360)/P14;
     desfase24=(((((ECap4Regs.CAP3-ECap4Regs.CAP2)*360)/(ECap4Regs.CAP3-
ECap4Regs.CAP1))));
    ECap4Regs.TSCTR=0;
}

void Frecuencia5()
{
    Uint32 P15;
    Uint32 duty15;
    Uint32 duty25;

    P15 = ECap5Regs.CAP3-ECap5Regs.CAP1; // Calculate 1st period
    Frec5 = 150e6/P15;

    duty15=ECap5Regs.CAP2-ECap5Regs.CAP1;
    duty25= ECap5Regs.CAP3-ECap5Regs.CAP2;
    desfase15=(duty25*360)/P15;
    desfase25=(((((ECap5Regs.CAP3-ECap5Regs.CAP2)*360)/(ECap5Regs.CAP3-
ECap5Regs.CAP1))));
   ECap5Regs.TSCTR=0;
}

void Frecuencia6()
{
    Uint32 P16;
    Uint32 duty16;
    Uint32 duty26;

     P16 = ECap6Regs.CAP3-ECap6Regs.CAP1; // Calculate 1st period
     Frec6 = 150e6/P16;

     duty16=ECap6Regs.CAP2-ECap6Regs.CAP1;
     duty26=ECap6Regs.CAP3-ECap6Regs.CAP2;
     desfase16=(duty26*360)/P16;
     desfase26=(((((ECap6Regs.CAP3-ECap6Regs.CAP2)*360)/(ECap6Regs.CAP3-
ECap6Regs.CAP1))));
    ECap6Regs.TSCTR=0;

}
```

11.8 CÓDIGO PRUEBA ENVIÓ Y RECEPCIÓN DE DATOS.

```
// ya recibe esta version
#include "DSP28x_Project.h"     // Device Headerfile and Examples Include File

extern Uint16 RamfuncsLoadStart;
extern Uint16 RamfuncsLoadEnd;
extern Uint16 RamfuncsRunStart;
// ciclo
void inicializacion(void);

// configuracion
void scib_init(void);// configura SCIB para recibir

// funciones

void CALCULOS();
void secuenciaenviar();


void divideval(int);

void tx_envia(char[]);
void rx_recibe(void);
void loop(void);

__interrupt void rxbint_isr(void);
__interrupt void cpu_timer0_isr(void);


int  tipo;
Uint16 numa,numa1,numa2,numa3,numa4,numa5;

int int1,int3,int2,int4,int5;


int condiok1;
int j=0;
Uint32 p=0;
Uint16 recibe2[2];

void tiempo(void);// funcion de tiempo
int seg=0;

Uint16 recibido=0;

int VVABRMS,VVBCRMS,VVCARMS;
int VIARMS,VIBRMS,VICRMS,VINRMS;

int VFrec=0, VFrec2=0, VFrec3=0, VFrec4=0,VFrec5=0, VFrec6=0;
int Vfase2=0,Vfase3=0;
int Vdesfase24=0, Vdesfase25=0, Vdesfase26=0;

int dia=23;
int mes=10;
int ano1=20;
```

```
int ano2=16;
int hora=12;
int min=30;
int nodo=1;
int YA=0;

void main(void)
{
 InitSysCtrl();

 InitSciaGpio();
 InitScibGpio();
 DINT;

  InitPieCtrl();
  IER = 0x0000;
  IFR = 0x0000;
 InitPieVectTable();
 MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd, &RamfuncsRunStart);
 InitFlash();
 //InitAdc();

 EALLOW;
 PieVectTable.SCIRXINTB=&rxbint_isr;
 PieVectTable.TINT0 = &cpu_timer0_isr;
 EDIS;    // This is needed to disable write to EALLOW protected registers

 InitCpuTimers();   // For this example, only initialize the Cpu Timers

 ConfigCpuTimer(&CpuTimer0, 150, 1000000);



  CpuTimer0Regs.TCR.all = 0x4000; // Use write-only instruction to set TSS bit =
0

  IER |= M_INT1;
  IER |=0x0100;  // interrupcion RX

    PieCtrlRegs.PIEIER9.bit.INTx3=1; // habilita interrupcion RX

// Enable global Interrupts and higher priority real-time debug events:
  EINT;   // Enable Global interrupt INTM
  ERTM;   // Enable Global realtime interrupt DBGM
  scib_init();
  //scia_init();

 inicializacion();

// Step 6. IDLE loop. Just sit and loop forever (optional):
   for(;;){

       if(YA==1)
       {
       secuenciaenviar();
```

```
        }
    }


}


void inicializacion(void)
{
    YA=0;
    recibido=0;

    recibe2[0]=0 ; recibe2[1]=0;
   // divideval(dato);
}

// configuraciones
void scib_init()
{
    ScibRegs.SCICCR.all =0x0007;            // 1 stop bit,  No loopback
                                                    // No parity,8 char
bits,
                                                    // async mode, idle-line
protocol
                ScibRegs.SCICTL1.all =0x0003;           // enable TX, RX,
internal SCICLK,
                                                // Disable RX ERR, SLEEP, TXWAKE
                ScibRegs.SCICTL2.all =0x0003;
                ScibRegs.SCICTL2.bit.RXBKINTENA = 1;
                ScibRegs.SCICTL2.bit.TXINTENA = 1;
                ScibRegs.SCIHBAUD    =0x0001;
                ScibRegs.SCILBAUD    =0x00E7;

                ScibRegs.SCIFFRX.all=0x0022;
                ScibRegs.SCIFFCT.all=0x0000;

                ScibRegs.SCICTL1.all =0x0023;     // Relinquish SCI from Reset

                //ScibRegs.SCIFFRX.all=0x204f;
                ScibRegs.SCIFFCT.all=0x0;
                ScibRegs.SCIFFTX.all=0xE040;
                //ScibRegs.SCIFFTX.bit.TXFIFOXRESET=1;
                ScibRegs.SCIFFRX.bit.RXFIFORESET=1;

}

//interrupciones
__interrupt void rxbint_isr(void)
{

    recibido = ScibRegs.SCIRXBUF.all;

        recibe2[r]=recibido;
        r++;
```

```
        if (r==2)
        {
            r=0;
        }

        if ((recibe2[0]==35 && recibe2[1]==42) )
        { y++;
            recibe2[0]=0; recibe2[1]=0;
        ScibRegs.SCIRXBUF.all=0;
                        CpuTimer0.InterruptCount=0;
                         min=0;
                         hora=0;
            PieCtrlRegs.PIEIER1.bit.INTx7 = 1;
            GpioDataRegs.GPADAT.bit.GPIO31=1;
            recibe2[0]=0; recibe2[1]=0;
           ScibRegs.SCIRXBUF.bit.RXDT=0;
        }
        if ((recibe2[0]==88 &&recibe2[1]==id) || (recibe2[0]==id
&&recibe2[1]==88))
           {
               GpioDataRegs.GPBDAT.bit.GPIO34  =0;
                     loop();
               GpioDataRegs.GPBDAT.bit.GPIO34 =1;
               //recibe2[0]=0; recibe2[1]=0;
              // ScibRegs.SCIRXBUF.bit.RXDT=0;
           }
     if( PieCtrlRegs.PIEIER1.bit.INTx7 == 1){
        if ((recibe2[0]==89 &&recibe2[1]==id) || (recibe2[0]==id
&&recibe2[1]==89))
               {
                 PET=1;
                 //ScibRegs.SCIRXBUF.bit.RXDT=0;
               }
     }
        PieCtrlRegs.PIEACK.all|=0x100;
        ScibRegs.SCIFFRX.bit.RXFFINTCLR=1;



    }


}
__interrupt void cpu_timer0_isr(void)
{
   seg=CpuTimer0.InterruptCount++;
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;

 if (seg==20)
 {
  CpuTimer0.InterruptCount=0;
  CALCULOS();
 }
}
// funciones
```

```
void divideval(int a)
{

// VALOR
    numa =a;
    numa1 = numa / 10000;
    numa2 = (numa % 10000)/1000;
    numa3 = (numa % 10000) % 1000 / 100;
    numa4 = ((numa % 10000) % 1000) % 100 / 10;
    numa5 = (((numa % 10000) % 1000) % 100) % 10;

    int1=numa1;
    int2=numa2;
    int3=numa3;
    int4=numa4;
    int5=numa5;



    char tx_buff1[] =
{60,nodo,tipo,dia,mes,ano1,ano2,hora,min,int1,int2,int3,int4,int5,62,0x13};


                do  {  tx_envia(tx_buff1); loop();


                }
                   while ((recibe2[0]!=condiok1 || recibe2[1]!=49)
&&(recibe2[0]!=49 || recibe2[1]!=condiok1) );


}

void tx_envia(char s[]){
     int k=0;

while (s[k] !=0x13)
{
    while (  ScibRegs.SCIFFTX.bit.TXFFST != 0) {}
          ScibRegs.SCITXBUF=s[k];
        loop();
        k++;
}
}
 void loop(void)
 {
     for(p=0;p<100000;p++)
     {
         ScibRegs.SCIFFRX.bit.RXFFINTCLR=1;
         ScibRegs.SCIRXBUF.all=0;
     }
 }

void CALCULOS()
```

```
{
                    VVABRMS= 121.2333*100;   VVBCRMS=121.4555*100;
VVCARMS=121.27333*100;
                    VIARMS =2.3222*100;   VIBRMS=2.432222*100;
VICRMS=2.76444*100;   VINRMS=2.52222*100;

                    VFrec=59.9965*100; VFrec2=59.9843*100; VFrec3=59.9934*100;
                    Vfase2=120.352*100;Vfase3=240.321*100;
                    Vdesfase24=30.23*100; Vdesfase25=10.324*100;
Vdesfase26=22.321*100;
                 YA=1;
}
void secuenciaenviar(){

    tipo=1; condiok1=65;
    divideval(VVABRMS);
    tipo=2; condiok1=66;
    divideval(VVBCRMS);
    tipo=3; condiok1=67;
    divideval(VVCARMS);
    tipo=4; condiok1=68;
    divideval(VIARMS);
    tipo=5; condiok1=69;
    divideval(VIBRMS);
    tipo=6; condiok1=70;
    divideval(VICRMS);
    tipo=7; condiok1=71;
    divideval(VINRMS);
    tipo=8; condiok1=72;
    divideval(VFrec);
    tipo=9; condiok1=73;
    divideval(VFrec2);
    tipo=10; condiok1=74;
    divideval(VFrec3);
    tipo=11; condiok1=75;
    divideval(Vfase2);
    tipo=12; condiok1=76;
    divideval(Vfase3);
    tipo=13; condiok1=77;
    divideval(Vdesfase24);
    tipo=14; condiok1=78;
    divideval(Vdesfase25);
    tipo=15; condiok1=79;
    divideval(Vdesfase26);
    inicializacion();
}
```

11.9 CODIGO NODO 1.

```
// esta version ya esta lista nodo 1


#include "DSP28x_Project.h"     // Device Headerfile and Examples Include File
#include <string.h>
#include <math.h>
```

```
#include <stdio.h>
#include <stdlib.h>

////VARIABLES RAM /////
extern Uint16 RamfuncsLoadStart;
extern Uint16 RamfuncsLoadEnd;
extern Uint16 RamfuncsRunStart;

////INTERRUPCIONES////
__interrupt void cpu_timer0_isr(void);
__interrupt void cpu_timer1_isr(void);

__interrupt void rxbint_isr(void);

__interrupt void ecap1_isr(void);
__interrupt void ecap2_isr(void);
__interrupt void ecap3_isr(void);

__interrupt void ecap4_isr(void);
__interrupt void ecap5_isr(void);
__interrupt void ecap6_isr(void);


////////CONFIGURACION////////
void ADCconfiguracion(void);
void ECAPconfiguracion(void);
void SCIBconfiguracion(void);



//////CICLO////////
void ADCLoop(void);

void Frecuencia1();
void Frecuencia2();
void Frecuencia3();
void Frecuencia4();
void Frecuencia5();
void Frecuencia6();

void inicializacion();
void tomadatos();
void CALCULOS();
void secuenciaenviar();

void divideval(int);

void tx_envia(char[]);
void rx_recibe(void);
void loop1(void);
void loop();

void tiempo(void);
```

```
//////VARIABLES CLK ADC///////
#define ADC_MODCLK 0x3 // HSPCLK = SYSCLKOUT/2*ADC_MODCLK2 = 150/(2*3)   = 25.0
MHz
#define ADC_CKPS   0x9   // ADC module clock = HSPCLK/2*ADC_CKPS   =
25.0MHz/(2*9) = 1.38MHz
#define ADC_SHCLK  0xf   // S/H VENTANA DEL PERIODO DEL MODULO ADC           =
16 ADC clocks

///FACTOR MULTIPLICATIVO CONVERSION 0-3V///
double multiplica=0.0007326007326007326;

//VECTORES
   //// VOLTAJES///
double MVAB[185],MVBC[185],MVCA[185];

  ////CORRIENTES///
double MIA[185],MIB[185],MIC[185],MINN[185];

   //VOLTAJES PROMEDIOS//
double MVAB2[20],MVBC2[20],MVCA2[20];

int y=0;
///VARIABLES PARA CICLOS
int j;
// VALORES DC Y RMS
  ////VOLTAJES
double FVABDC,FVBCDC,FVCADC;
double FVABRMS,FVBCRMS,FVCARMS;
double FVABRMS2,FVBCRMS2,FVCARMS2; // raices
double VABRMS,VBCRMS,VCARMS; // raiz final para enviar
/// PROMEDIO DE RMS
double PVABRMS,PVBCRMS,PVCARMS;

  ////CORRIENTES
double FIADC,FIBDC,FICDC,FINDC;
double FIARMS,FIBRMS,FICRMS,FINRMS;
double FIARMS2,FIBRMS2,FICRMS2,FINRMS2;// raices
double IARMS,IBRMS,ICRMS,INRMS;// raiz final para enviar


///VARIABLES ECAP///

// variable activacion y desactivacion //
int true; // FASES
int YA=0; // ENVIO POR MIN
int PET=0;
   /// frecuencias//
double Frec, Frec2, Frec3,Frec4,Frec5,Frec6;
 /// contafores para fases//
double contador; //TIMER1
double contador1; // FASE2
double contador2; // FASE3
 /// fases//
double fase2,fase3;
```

```
 /// angulos //

double desfase24, desfase25, desfase26;

double fin =5.333;

// valores a enviar
int VVABRMS,VVBCRMS,VVCARMS;
int VIARMS,VIBRMS,VICRMS,VINRMS;

int VFrec, VFrec2, VFrec3, VFrec4,VFrec5, VFrec6;
int Vfase2,Vfase3;
int Vdesfase24, Vdesfase25, Vdesfase26;

// variables tiempo
int dia=6;
int mes=11;
int ano1;
int ano2;
Uint16 ano=2016;
int hora=0;
int min=0;
int seg=0;


// variables envio y recibido
int r=0;
int  tipo;
int nodo=1;
int id=49;
int M=0;
int condiok1;
int idinicio=60,idfinal=62;
Uint16 recibe2[2];
Uint16 recibido=0;

Uint16 numa,numa1,numa2,numa3,numa4,numa5,numano1,numano2;
  int int1,int3,int2,int4,int5;

 Uint32 e;
void main(void)
{
      InitSysCtrl();
   // frecuencias  voltajes//
      InitECap1Gpio();
      InitECap2Gpio();
      InitECap3Gpio();
   // angulos ///
      InitECap4Gpio();
      InitECap5Gpio();
      InitECap6Gpio();

      //sci
      InitSciaGpio();
      InitScibGpio();
```

```
        EALLOW;
        SysCtrlRegs.HISPCP.all = ADC_MODCLK;    // HSPCLK = SYSCLKOUT/ADC_MODCLK
        EDIS;
        DINT;

   InitPieCtrl();

   IER = 0x0000;
   IFR = 0x0000;

   InitPieVectTable();
   MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd, &RamfuncsRunStart);
   EALLOW;  // This is needed to write to EALLOW protected registers
     PieVectTable.ECAP1_INT = &ecap1_isr;
     PieVectTable.ECAP2_INT = &ecap2_isr;
     PieVectTable.ECAP3_INT = &ecap3_isr;
     PieVectTable.ECAP4_INT = &ecap4_isr;
     PieVectTable.ECAP5_INT = &ecap5_isr;
     PieVectTable.ECAP6_INT = &ecap6_isr;
     PieVectTable.SCIRXINTB=&rxbint_isr;
     PieVectTable.TINT0 = &cpu_timer0_isr;
     PieVectTable.XINT13 = &cpu_timer1_isr;
     GpioCtrlRegs.GPBMUX1.bit.GPIO34 = 0;    // 0=GPIO,  1=ECAP1,  2=Resv,
3=Resv
     GpioCtrlRegs.GPBDIR.bit.GPIO34 = 1;
     GpioCtrlRegs.GPAMUX2.bit.GPIO31 = 0;    // 0=GPIO,  1=ECAP1,  2=Resv,
3=Resv
     GpioCtrlRegs.GPADIR.bit.GPIO31 = 1;

     EDIS;
     InitCpuTimers();   // For this example, only initialize the Cpu Timers


         ConfigCpuTimer(&CpuTimer0, 150,1000000);
         ConfigCpuTimer(&CpuTimer1, 150,1);


           CpuTimer0Regs.TCR.all = 0x4000; // Use write-only instruction to set
TSS bit = 0
           CpuTimer1Regs.TCR.all = 0x4000; // Use write-only instruction to set
TSS bit = 0


   InitFlash();
   InitAdc();



     IER |= M_INT4;  ///ECAP
     IER |= M_INT1;  //TIMER0
     IER |= M_INT13; //TIMER1
     IER |=0x0100;  // interrupcion RX

     CpuTimer1Regs.TCR.bit.TIE = 0;
```

```
      PieCtrlRegs.PIEIER9.bit.INTx3=1; // RX
      GpioDataRegs.GPBDAT.bit.GPIO34=1; // LED APAGADO
     // GpioDataRegs.GPADAT.bit.GPIO31=0; //
    //  PieCtrlRegs.PIEIER1.bit.INTx7 = 1;

      EINT;
      ERTM;
          inicializacion();
          SCIBconfiguracion();
          ADCconfiguracion();
          ECAPconfiguracion();


    for(;;){
        if(YA==1)
               {
               tomadatos();
               secuenciaenviar();

               }
        if(PET==1)
                    {
                    tomadatos();
                    secuenciaenviar();

                    }

    }

}
////////////////////////////////////////////////////////////////////////////
////////INTERRUPCIONES//////////////
__interrupt void rxbint_isr(void)
{

     recibido = ScibRegs.SCIRXBUF.all;

     recibe2[r]=recibido;
     r++;
     if (r==2)
     {
         r=0;
     }

     if ((recibe2[0]==35 && recibe2[1]==42) )
     { y++;
         recibe2[0]=0; recibe2[1]=0;
     ScibRegs.SCIRXBUF.all=0;
                    CpuTimer0.InterruptCount=0;
                     min=0;
                     hora=0;
         PieCtrlRegs.PIEIER1.bit.INTx7 = 1;

         recibe2[0]=0; recibe2[1]=0;
        ScibRegs.SCIRXBUF.bit.RXDT=0;
```

```
    }
    if ((recibe2[0]==88 &&recibe2[1]==id) || (recibe2[0]==id &&recibe2[1]==88))
       {
           GpioDataRegs.GPBDAT.bit.GPIO34  =0;
                 loop();
           GpioDataRegs.GPBDAT.bit.GPIO34 =1;
           //recibe2[0]=0; recibe2[1]=0;
          // ScibRegs.SCIRXBUF.bit.RXDT=0;
       }
 if( PieCtrlRegs.PIEIER1.bit.INTx7 == 1){
    if ((recibe2[0]==89 &&recibe2[1]==id) || (recibe2[0]==id &&recibe2[1]==89))
           {
             PET=1;
             //ScibRegs.SCIRXBUF.bit.RXDT=0;
           }
 }
    PieCtrlRegs.PIEACK.all|=0x100;
    ScibRegs.SCIFFRX.bit.RXFFINTCLR=1;



}

__interrupt void cpu_timer0_isr(void)
{
  seg=CpuTimer0.InterruptCount++;
  tiempo();

   PieCtrlRegs.PIEACK.all = PIEACK_GROUP1;

if((min==1||min==5||min==9||min==13||min==17||min==21||min==25||min==29||min==33
||min==37||min==41||min==45||min==49||min==53||min==57) && seg==0){
 //
if((min==3||min==7||min==11||min==15||min==19||min==23||min==27||min==31||min==3
5||min==39||min==43||min==47||min==51||min==55||min==59) && seg==0){
       YA=1;
   }


}

__interrupt void cpu_timer1_isr(void)
{
  contador= CpuTimer1.InterruptCount++;
   EDIS;
}


__interrupt void ecap1_isr(void)
{
   true++;

    if (true<3)
    {
   CpuTimer1Regs.TCR.bit.TIE = 1;
```

```
   PieCtrlRegs.PIEIER4.bit.INTx2 = 1;
        contador1=contador;
    }

    if (true>3 && true<6)
       {
      CpuTimer1Regs.TCR.bit.TIE = 0;
      PieCtrlRegs.PIEIER4.bit.INTx2 = 0;
      CpuTimer1.InterruptCount=0;
        }
    if (true>6 && true<9)
    {
        CpuTimer1Regs.TCR.bit.TIE = 1;
        PieCtrlRegs.PIEIER4.bit.INTx3 = 1;
        contador2=contador;
    }
    if (true>9 && true<11)
        {
       CpuTimer1Regs.TCR.bit.TIE = 0;
       PieCtrlRegs.PIEIER4.bit.INTx3 = 0;
       CpuTimer1.InterruptCount=0;
         }

    if (true>12)
            {

           PieCtrlRegs.PIEIER4.bit.INTx1 = 0;
           PieCtrlRegs.PIEIER4.bit.INTx2 = 0;
           PieCtrlRegs.PIEIER4.bit.INTx3 = 0;
           PieCtrlRegs.PIEIER4.bit.INTx4 = 0;
           PieCtrlRegs.PIEIER4.bit.INTx5 = 0;
           PieCtrlRegs.PIEIER4.bit.INTx6 = 0;
           CpuTimer1Regs.TCR.bit.TIE = 0;
           CpuTimer1.InterruptCount=0;
           true=0;
             }

       Frecuencia1();

   ECap1Regs.ECCLR.bit.CEVT4 = 1;
   ECap1Regs.ECCLR.bit.INT = 1;
   ECap1Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;
   fase2=((contador1-54895)/(-405.325))+120;
   fase3=((contador2-54895)/(-405.325))+120;
}

__interrupt void ecap2_isr(void)
{
    CpuTimer1.InterruptCount=0;
    Frecuencia2();
   ECap2Regs.ECCLR.bit.CEVT4 = 1;
   ECap2Regs.ECCLR.bit.INT = 1;
```

```
   ECap2Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;

}

__interrupt void ecap3_isr(void)
{
    CpuTimer1.InterruptCount=0;
    Frecuencia3();
   ECap3Regs.ECCLR.bit.CEVT4 = 1;
   ECap3Regs.ECCLR.bit.INT = 1;
   ECap3Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;

 }


__interrupt void ecap4_isr(void)
{
    Frecuencia4();
   ECap4Regs.ECCLR.bit.CEVT4 = 1;
   ECap4Regs.ECCLR.bit.INT = 1;
   ECap4Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;
}
__interrupt void ecap5_isr(void)
{

    Frecuencia5();
   ECap5Regs.ECCLR.bit.CEVT4 = 1;
   ECap5Regs.ECCLR.bit.INT = 1;
   ECap5Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;
}

__interrupt void ecap6_isr(void)
{

    Frecuencia6();
   ECap6Regs.ECCLR.bit.CEVT4 = 1;
   ECap6Regs.ECCLR.bit.INT = 1;
   ECap6Regs.ECCTL2.bit.REARM = 1;

   // Acknowledge this interrupt to receive more interrupts from group 4
   PieCtrlRegs.PIEACK.all = PIEACK_GROUP4;
}
////////////////////////////////////////////////////////////////////////////
```

```
////////CONFIGURACIONES//////////////

void SCIBconfiguracion()
{
    ScibRegs.SCICCR.all =0x0007;            // 1 stop bit,  No loopback
                                                        // No parity,8 char
bits,
                                                        // async mode, idle-line
protocol
                ScibRegs.SCICTL1.all =0x0003;           // enable TX, RX,
internal SCICLK,
                                                // Disable RX ERR, SLEEP, TXWAKE
                ScibRegs.SCICTL2.all =0x0003;
                ScibRegs.SCICTL2.bit.RXBKINTENA = 1;
                ScibRegs.SCICTL2.bit.TXINTENA = 1;
                ScibRegs.SCIHBAUD    =0x0001;
                ScibRegs.SCILBAUD    =0x00E7;

                ScibRegs.SCIFFRX.all=0x0022;
                ScibRegs.SCIFFCT.all=0x0000;

                ScibRegs.SCICTL1.all =0x0023;     // Relinquish SCI from Reset

                //ScibRegs.SCIFFRX.all=0x204f;
                ScibRegs.SCIFFCT.all=0x0;
                ScibRegs.SCIFFTX.all=0xE040;
                //ScibRegs.SCIFFTX.bit.TXFIFOXRESET=1;
                ScibRegs.SCIFFRX.bit.RXFIFORESET=1;

}


void ADCconfiguracion(void)
{

         AdcRegs.ADCTRL1.bit.ACQ_PS = ADC_SHCLK;  //
         AdcRegs.ADCTRL3.bit.ADCCLKPS = ADC_CKPS; //

         AdcRegs.ADCTRL2.bit.SOC_SEQ1 = 1;        //
         AdcRegs.ADCTRL3.bit.SMODE_SEL = 1;       //
         AdcRegs.ADCTRL1.bit.SEQ_CASC = 1;        //

         AdcRegs.ADCTRL2.bit.INT_ENA_SEQ1= 1;     //
         AdcRegs.ADCTRL2.bit.RST_SEQ1 = 1;        //

         AdcRegs.ADCCHSELSEQ1.bit.CONV00 = 0x0;   // A0-B0
         AdcRegs.ADCCHSELSEQ1.bit.CONV01 = 0x1;   // A1-B1
         AdcRegs.ADCCHSELSEQ1.bit.CONV02 = 0x2;   // A2-B2
         AdcRegs.ADCCHSELSEQ1.bit.CONV03 = 0x3;   // A3-B3
         AdcRegs.ADCCHSELSEQ1.bit.CONV03 = 0x3;   //
         AdcRegs.ADCCHSELSEQ2.bit.CONV04 = 0x4;   //
         AdcRegs.ADCCHSELSEQ2.bit.CONV05 = 0x5;   //


         AdcRegs.ADCMAXCONV.bit.MAX_CONV1=0x0006; //
```

```
}

void ECAPconfiguracion(void)
{
    //ECAP1   configura fases y frecuencias

    //ECAP4
        ECap1Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
        ECap1Regs.ECCLR.all = 0xFFFF;              // Clear all CAP interrupt
flags
        ECap1Regs.ECCTL1.bit.CAPLDEN = 0;          // Disable CAP1-CAP4 register
loads
        ECap1Regs.ECCTL2.bit.TSCTRSTOP = 0;        // Make sure the counter is
stopped
        // Configure peripheral registers
        ECap1Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
        ECap1Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
        ECap1Regs.ECCTL1.bit.CAP1POL = 0;          // Rising edge
        ECap1Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
        ECap1Regs.ECCTL1.bit.CAP3POL = 0;          // Rising edge
        ECap1Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
        ECap1Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
        ECap1Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
        ECap1Regs.ECCTL1.bit.CTRRST3 = 0;          // Difference operation
        ECap1Regs.ECCTL1.bit.CTRRST4 = 0;          // Difference operation
        ECap1Regs.ECCTL2.bit.SYNCI_EN = 1;         // Enable sync in
        ECap1Regs.ECCTL2.bit.SYNCO_SEL = 0;        // Pass through
        ECap1Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable capture units


        ECap1Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
        ECap1Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
        ECap1Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4 register
loads
        ECap1Regs.ECEINT.bit.CEVT4 = 1;            // 4 events = interrupt

       // cap5
        ECap2Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
        ECap2Regs.ECCLR.all = 0xFFFF;              // Clear all CAP interrupt
flags
        ECap2Regs.ECCTL1.bit.CAPLDEN = 0;          // Disable CAP1-CAP4 register
loads
        ECap2Regs.ECCTL2.bit.TSCTRSTOP = 0;        // Make sure the counter is
stopped
        // C2nfigure peripheral registers
        ECap2Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
        ECap2Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
        ECap2Regs.ECCTL1.bit.CAP1POL = 0;          // Falling edge
        ECap2Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
        ECap2Regs.ECCTL1.bit.CAP3POL = 0;          // Falling edge
        ECap2Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
        ECap2Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
        ECap2Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
```

```
        ECap2Regs.ECCTL1.bit.CTRRST3 = 0;           // Difference operation
        ECap2Regs.ECCTL1.bit.CTRRST4 = 0;           // Difference operation
        ECap2Regs.ECCTL2.bit.SYNCI_EN = 1;          // Enable sync in
        ECap2Regs.ECCTL2.bit.SYNCO_SEL = 0;         // Pass through
        ECap2Regs.ECCTL1.bit.CAPLDEN = 1;           // Enable capture units


        ECap2Regs.ECCTL2.bit.TSCTRSTOP = 1;         // Start Counter
        ECap2Regs.ECCTL2.bit.REARM = 1;             // arm one-shot
        ECap2Regs.ECCTL1.bit.CAPLDEN = 1;           // Enable CAP1-CAP4 register
loads
        ECap2Regs.ECEINT.bit.CEVT4 = 1;             // 4 events = interrupt

          //cap3
        ECap3Regs.ECEINT.all = 0x0000;              // Disable all capture
interrupts
        ECap3Regs.ECCLR.all = 0xFFFF;               // Clear all CAP interrupt
flags
        ECap3Regs.ECCTL1.bit.CAPLDEN = 0;           // Disable CAP1-CAP4 register
loads
        ECap3Regs.ECCTL2.bit.TSCTRSTOP = 0;         // Make sure the counter is
stopped
         // Configure peripheral registers
        ECap3Regs.ECCTL2.bit.CONT_ONESHT = 1;       // One-shot
        ECap3Regs.ECCTL2.bit.STOP_WRAP = 3;         // Stop at 4 events
        ECap3Regs.ECCTL1.bit.CAP1POL = 0;           // Falling edge
        ECap3Regs.ECCTL1.bit.CAP2POL = 0;           // Rising edge
        ECap3Regs.ECCTL1.bit.CAP3POL = 0;           // Falling edge
        ECap3Regs.ECCTL1.bit.CAP4POL = 0;           // Rising edge
        ECap3Regs.ECCTL1.bit.CTRRST1 = 0;           // Difference operation
        ECap3Regs.ECCTL1.bit.CTRRST2 = 0;           // Difference operation
        ECap3Regs.ECCTL1.bit.CTRRST3 = 0;           // Difference operation
        ECap3Regs.ECCTL1.bit.CTRRST4 = 0;           // Difference operation
        ECap3Regs.ECCTL2.bit.SYNCI_EN = 1;          // Enable sync in
        ECap3Regs.ECCTL2.bit.SYNCO_SEL = 0;         // Pass through
        ECap3Regs.ECCTL1.bit.CAPLDEN = 1;           // Enable capture units

        ECap3Regs.ECCTL2.bit.TSCTRSTOP = 1;         // Start Counter
        ECap3Regs.ECCTL2.bit.REARM = 1;             // arm one-shot
        ECap3Regs.ECCTL1.bit.CAPLDEN = 1;           // Enable CAP1-CAP4 register
loads
        ECap3Regs.ECEINT.bit.CEVT4 = 1;             // 4 events = interrupt
// 4 events = interrupt

//// configura para desfases y frecuencias
    //ECAP4

          ECap4Regs.ECEINT.all = 0x0000;              // Disable all capture
interrupts
          ECap4Regs.ECCLR.all = 0xFFFF;               // Clear all CAP interrupt
flags
          ECap4Regs.ECCTL1.bit.CAPLDEN = 0;           // Disable CAP1-CAP4
register loads
          ECap4Regs.ECCTL2.bit.TSCTRSTOP = 0;         // Make sure the counter is
stopped
```

```
          // Configure peripheral registers
          ECap4Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
          ECap4Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
          ECap4Regs.ECCTL1.bit.CAP1POL = 1;          // Falling edge
          ECap4Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
          ECap4Regs.ECCTL1.bit.CAP3POL = 1;          // Falling edge
          ECap4Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
          ECap4Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
          ECap4Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
          ECap4Regs.ECCTL1.bit.CTRRST3 = 0;          // Difference operation
          ECap4Regs.ECCTL1.bit.CTRRST4 = 0;          // Difference operation
          ECap4Regs.ECCTL2.bit.SYNCI_EN = 1;         // Enable sync in
          ECap4Regs.ECCTL2.bit.SYNCO_SEL = 0;        // Pass through
          ECap4Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable capture units


          ECap4Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
          ECap4Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
          ECap4Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4
register loads
          ECap4Regs.ECEINT.bit.CEVT4 = 1;
   // cap5
      ECap5Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
      ECap5Regs.ECCLR.all = 0xFFFF;              // Clear all CAP interrupt
flags
      ECap5Regs.ECCTL1.bit.CAPLDEN = 0;          // Disable CAP1-CAP4 register
loads
      ECap5Regs.ECCTL2.bit.TSCTRSTOP = 0;        // Make sure the counter is
stopped
      // Configure peripheral registers
      ECap5Regs.ECCTL2.bit.CONT_ONESHT = 1;      // One-shot
      ECap5Regs.ECCTL2.bit.STOP_WRAP = 3;        // Stop at 4 events
      ECap5Regs.ECCTL1.bit.CAP1POL = 1;          // Falling edge
      ECap5Regs.ECCTL1.bit.CAP2POL = 0;          // Rising edge
      ECap5Regs.ECCTL1.bit.CAP3POL = 1;          // Falling edge
      ECap5Regs.ECCTL1.bit.CAP4POL = 0;          // Rising edge
      ECap5Regs.ECCTL1.bit.CTRRST1 = 0;          // Difference operation
      ECap5Regs.ECCTL1.bit.CTRRST2 = 0;          // Difference operation
      ECap5Regs.ECCTL1.bit.CTRRST3 = 0;          // Difference operation
      ECap5Regs.ECCTL1.bit.CTRRST4 = 0;          // Difference operation
      ECap5Regs.ECCTL2.bit.SYNCI_EN = 1;         // Enable sync in
      ECap5Regs.ECCTL2.bit.SYNCO_SEL = 0;        // Pass through
      ECap5Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable capture units


      ECap5Regs.ECCTL2.bit.TSCTRSTOP = 1;        // Start Counter
      ECap5Regs.ECCTL2.bit.REARM = 1;            // arm one-shot
      ECap5Regs.ECCTL1.bit.CAPLDEN = 1;          // Enable CAP1-CAP4 register
loads
      ECap5Regs.ECEINT.bit.CEVT4 = 1;            // 4 events = interrupt

      //cap6
      ECap6Regs.ECEINT.all = 0x0000;             // Disable all capture
interrupts
```

```
      ECap6Regs.ECCLR.all = 0xFFFF;             // Clear all CAP interrupt
flags
      ECap6Regs.ECCTL1.bit.CAPLDEN = 0;         // Disable CAP1-CAP4 register
loads
      ECap6Regs.ECCTL2.bit.TSCTRSTOP = 0;       // Make sure the counter is
stopped
      // Configure peripheral registers
      ECap6Regs.ECCTL2.bit.CONT_ONESHT = 1;     // One-shot
      ECap6Regs.ECCTL2.bit.STOP_WRAP = 3;       // Stop at 4 events
      ECap6Regs.ECCTL1.bit.CAP1POL = 1;         // Falling edge
      ECap6Regs.ECCTL1.bit.CAP2POL = 0;         // Rising edge
      ECap6Regs.ECCTL1.bit.CAP3POL = 1;         // Falling edge
      ECap6Regs.ECCTL1.bit.CAP4POL = 0;         // Rising edge
      ECap6Regs.ECCTL1.bit.CTRRST1 = 0;         // Difference operation
      ECap6Regs.ECCTL1.bit.CTRRST2 = 0;         // Difference operation
      ECap6Regs.ECCTL1.bit.CTRRST3 = 0;         // Difference operation
      ECap6Regs.ECCTL1.bit.CTRRST4 = 0;         // Difference operation
      ECap6Regs.ECCTL2.bit.SYNCI_EN = 1;        // Enable sync in
      ECap6Regs.ECCTL2.bit.SYNCO_SEL = 0;       // Pass through
      ECap6Regs.ECCTL1.bit.CAPLDEN = 1;         // Enable capture units


      ECap6Regs.ECCTL2.bit.TSCTRSTOP = 1;       // Start Counter
      ECap6Regs.ECCTL2.bit.REARM = 1;           // arm one-shot
      ECap6Regs.ECCTL1.bit.CAPLDEN = 1;         // Enable CAP1-CAP4 register
loads
      ECap6Regs.ECEINT.bit.CEVT4 = 1;           // 4 events = interrupt
}

//////////////////////////////////////////////////////////////////////////////
////////CICLOS//////////////

void inicializacion(){


      int i;
      idinicio=60; idfinal=62;
     /// LIMPIEZA VARIABLES Y VECTORES//
       for (i=0; i<185; i++)
       { YA=0;
         PET=0;
         j=0;

         MVAB[i] = 0;  MVBC[i] = 0; MVCA[i] = 0;
         MIA[i] = 0;   MIB[i] = 0;  MIC[i] = 0 , MINN[i] = 0;

         FVABDC=0;   FVBCDC=0;   FVCADC=0;
         FVABRMS=0;  FVBCRMS=0;  FVCARMS=0;

         FIADC=0;    FIBDC=0;   FICDC=0; FINDC=0;
         FIARMS=0;   FIBRMS=0;  FICRMS=0; FINRMS=0;

     true=0;
         /// frecuencias//
     Frec=0; Frec2=0; Frec3=0; Frec4=0; Frec5=0; Frec6=0;
```

```
      /// contafores para fases//
    contador=0; //TIMER1
    contador1=0; // FASE2
    contador2=0; // FASE3
      /// fases//
    fase2=0;fase3=0;
      /// angulos //

    desfase24=0; desfase25=0; desfase26=0;

     VVABRMS=0;VVBCRMS=0;VVCARMS=0;
     VIARMS=0;VIBRMS=0;VICRMS=0;VINRMS=0;

     VFrec=0; VFrec2=0; VFrec3=0; VFrec4=0;VFrec5=0; VFrec6=0;
     Vfase2=0;Vfase3=0;
     Vdesfase24=0; Vdesfase25=0; Vdesfase26=0;
    // r=0;
     recibido=0;
     recibe2[0]=0 ; recibe2[1]=0;



      }

}
void tomadatos(){
      int k;

       for (k=0; k<20; k++)
      {
                ADCLoop();
                /// interrupciones ecap
                 PieCtrlRegs.PIEIER4.bit.INTx1 = 1;

                 PieCtrlRegs.PIEIER4.bit.INTx4 = 1;
                 PieCtrlRegs.PIEIER4.bit.INTx5 = 1;
                 PieCtrlRegs.PIEIER4.bit.INTx6 = 1;

      }
            CALCULOS();


}

void ADCLoop(void)
{

int i;
 /// LIMPIEZA VARIABLES Y VECTORES//
   for (i=0; i<185; i++)
   {
     MVAB[i] = 0;  MVBC[i] = 0; MVCA[i] = 0;
     MIA[i] = 0;   MIB[i] = 0;  MIC[i] = 0 , MINN[i] = 0;

     FVABDC=0;   FVBCDC=0;   FVCADC=0;
```

```
     FVABRMS=0;   FVBCRMS=0;   FVCARMS=0;

     FIADC=0;     FIBDC=0;    FICDC=0; FINDC=0;
     FIARMS=0;    FIBRMS=0;   FICRMS=0; FINRMS=0;
   }
//// CAPTURA DE DATOS ////

    for (i=0; i<185; i++)
     {
          AdcRegs.ADCTRL2.bit.SOC_SEQ1 = 1;
          MVAB[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT0>>4))*multiplica))-
1.653271)/0.003272)+50.1);
          MVBC[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT2>>4))*multiplica))-
1.599372)/0.003044)+49.8);
          MVCA[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT4>>4))*multiplica))-
1.726426)/0.003038)+49.7);

          MIA[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT1>>4))*multiplica))-
1.594349)/0.021908)+2.1);
          MIB[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT3>>4))*multiplica))-
1.585767)/0.021340)+2.14);
          MIC[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT5>>4))*multiplica))-
1.591941)/0.021034)+2.14);
          MINN[i]=sqrt(2)*(((((((AdcRegs.ADCRESULT7>>4))*multiplica))-
1.513762)/0.019108));

         while (AdcRegs.ADCST.bit.INT_SEQ1== 0) {} // Wait for interrupt
          AdcRegs.ADCST.bit.INT_SEQ1_CLR = 1;
     }


/// VALORES DC Y RMS ///

   for (i=1;i<185;i++)
  {
      FVABDC=(MVAB[i]/184)+FVABDC; FVBCDC=(MVBC[i]/184)+FVBCDC;
FVCADC=(MVCA[i]/184)+FVCADC;
      FIADC=(MIA[i]/184)+FIADC; FIBDC=(MIB[i]/184)+FIBDC;
FICDC=(MIC[i]/184)+FICDC;

      FVABRMS=(((MVAB[i]*MVAB[i])/184)+FVABRMS);
FVBCRMS=(((MVBC[i]*MVBC[i])/184)+FVBCRMS);
FVCARMS=(((MVCA[i]*MVCA[i])/184)+FVCARMS);
      FIARMS=(((MIA[i]*MIA[i])/184)+FIARMS);
FIBRMS=(((MIB[i]*MIB[i])/184)+FIBRMS);FICRMS=(((MIC[i]*MIC[i])/184)+FICRMS);FINR
MS=(((MINN[i]*MINN[i])/184)+FINRMS);
  }

   FVABRMS2=sqrt(FVABRMS);    FVBCRMS2=sqrt(FVBCRMS);    FVCARMS2=sqrt(FVCARMS);
//
   // RAIZ FINAL RMS-DC
   VABRMS=sqrt((FVABRMS2*FVABRMS2)-(FVABDC*FVABDC));
VBCRMS=sqrt((FVBCRMS2*FVBCRMS2)-(FVBCDC*FVBCDC));
VCARMS=sqrt((FVCARMS2*FVCARMS2)-(FVCADC*FVCADC));
```

```
   MVAB2[j]=VABRMS;  MVBC2[j]=VCARMS;  MVCA2[j]=VCARMS;

   FIARMS2=sqrt(FIARMS); FIBRMS2=sqrt(FIBRMS); FICRMS2=sqrt(FICRMS);
FINRMS2=sqrt(FINRMS);

   IARMS=sqrt((FIARMS2*FIARMS2)-(FIADC-FIADC)); IBRMS=sqrt((FIBRMS2*FIBRMS2)-
(FIBDC-FIBDC)); ICRMS=sqrt((FICRMS2*FICRMS2)-(FICDC-FICDC));
INRMS=sqrt((FINRMS2*FINRMS2)-(FIADC-FINDC));

   j++;
// promedio VOLTAJES //
     if (j==20)
    {
        PVABRMS=( MVAB2[0]+ MVAB2[1]+ MVAB2[2]+ MVAB2[3] + MVAB2[4]+ MVAB2[5]+
MVAB2[6]+ MVAB2[7]+ MVAB2[8]+ MVAB2[9]+
                MVAB2[10]+ MVAB2[11]+ MVAB2[12]+ MVAB2[13] + MVAB2[14]+
MVAB2[15]+ MVAB2[16]+ MVAB2[17]+ MVAB2[18]+ MVAB2[19])/20;
        PVBCRMS=( MVBC2[0]+ MVBC2[1]+ MVBC2[2]+ MVBC2[3] + MVBC2[4]+ MVBC2[5]+
MVBC2[6]+ MVBC2[7]+ MVBC2[8]+ MVBC2[9]+
                        MVBC2[10]+ MVBC2[11]+ MVBC2[12]+ MVBC2[13] + MVBC2[14]+
MVBC2[15]+ MVBC2[16]+ MVBC2[17]+ MVBC2[18]+ MVBC2[19])/20;
        PVCARMS=( MVCA2[0]+ MVCA2[1]+ MVCA2[2]+ MVCA2[3] + MVCA2[4]+ MVCA2[5]+
MVCA2[6]+ MVCA2[7]+ MVCA2[8]+ MVCA2[9]+
                                MVCA2[10]+ MVCA2[11]+ MVCA2[12]+ MVCA2[13] +
MVCA2[14]+ MVCA2[15]+ MVCA2[16]+ MVCA2[17]+ MVCA2[18]+ MVCA2[19])/20;

    }


 }

void Frecuencia1()
{
    Uint32 P1;
    Uint32 P2;

    P1 = ECap1Regs.CAP2-ECap1Regs.CAP1; // Calculate 1st period
    P2 = ECap1Regs.CAP3-ECap1Regs.CAP2; // Calculate 2st period

        if(P1<P2)
        {
           Frec = 150e6/P1;
        }
        else if (P2<P1)
        {
           Frec = 150e6/P2;
        }

        ECap1Regs.TSCTR=0;

}

void Frecuencia2()
{
    Uint32 P12;
```

```
    Uint32 P22;

    P12 = ECap2Regs.CAP2-ECap2Regs.CAP1; // Calculate 1st period
    P22 = ECap2Regs.CAP3-ECap2Regs.CAP2; // Calculate 2st period


    if(P12<P22)
    {
       Frec2 = 150e6/P12;
    }
    else if (P22<P12)
    {
       Frec2 = 150e6/P22;
    }

    ECap2Regs.TSCTR=0;
}

void Frecuencia3()
{
    Uint32 P13;
    Uint32 P23;

    P13 = ECap3Regs.CAP2-ECap3Regs.CAP1; // Calculate 1st period
    P23 = ECap3Regs.CAP3-ECap3Regs.CAP2; // Calculate 2st period

    if(P13<P23)
    {
       Frec3 = 150e6/P13;
    }
    else if (P23<P13)
    {
       Frec3 = 150e6/P23;
    }
    ECap3Regs.TSCTR=0;
}
//// desfases
void Frecuencia4()
{
        Uint32 P14;
        Uint32 duty14;
        Uint32 duty24;

         P14 =  ECap4Regs.CAP3-ECap4Regs.CAP1; // Calculate 1st period
         Frec4 = 150e6/P14;

         duty14=ECap4Regs.CAP2-ECap4Regs.CAP1;
         duty24= ECap4Regs.CAP3-ECap4Regs.CAP2;
        // desfase14=(duty14*360)/P14;
         desfase24=((((duty24*360)/P14)));
        ECap4Regs.TSCTR=0;
}
// defase
void Frecuencia5()
{
```

```
    Uint32 P15;
       Uint32 duty15;
       Uint32 duty25;

       P15 = ECap5Regs.CAP3-ECap5Regs.CAP1; // Calculate 1st period
       Frec5 = 150e6/P15;

       duty15=ECap5Regs.CAP2-ECap5Regs.CAP1;
       duty25=ECap5Regs.CAP3-ECap5Regs.CAP2;
     //  desfase15=(duty15*360)/P15;
       desfase25=((((duty25*360)/P15)));
       ECap5Regs.TSCTR=0;
}

void Frecuencia6()
{
       Uint32 P16;
       Uint32 duty16;
       Uint32 duty26;

        P16 = ECap6Regs.CAP3-ECap6Regs.CAP1; // Calculate 1st period
        Frec6 = 150e6/P16;

        duty16=ECap6Regs.CAP2-ECap6Regs.CAP1;
        duty26=ECap6Regs.CAP3-ECap6Regs.CAP2;
        //desfase16=(duty16*360)/P16;
        desfase26=((((duty26*360)/P16)));
       ECap6Regs.TSCTR=0;
}



void divideval(int a)
{

// VALOR
    numa =a;
    numa1 = numa / 10000;
    numa2 = (numa % 10000)/1000;
    numa3 = (numa % 10000) % 1000 / 100;
    numa4 = ((numa % 10000) % 1000) % 100 / 10;
    numa5 = (((numa % 10000) % 1000) % 100) % 10;

    int1=numa1;
    int2=numa2;
    int3=numa3;
    int4=numa4;
    int5=numa5;
    numano1=ano / 100;
    numano2=(ano % 100);;
    ano1=numano1;
    ano2=numano2;
```

```
    char tx_buff1[] =
{idinicio,nodo,tipo,dia,mes,ano1,ano2,hora,min,int1,int2,int3,int4,int5,idfinal,
0x777};


                do  {  tx_envia(tx_buff1); loop1();


                }
                   while ((recibe2[0]!=condiok1 || recibe2[1]!=id)
&&(recibe2[0]!=id || recibe2[1]!=condiok1) );


}

void tx_envia(char s[]){
     int k=0;

while (s[k] !=0x777)
{
    while (  ScibRegs.SCIFFTX.bit.TXFFST != 0) {}
          ScibRegs.SCITXBUF=s[k];
        loop1();
        k++;
}
}
 void loop1(void)
 {
     for(e=0;e<100000;e++)
     {
         ScibRegs.SCIFFRX.bit.RXFFINTCLR=1;
         ScibRegs.SCIRXBUF.all=0;
     }
 }

 void loop()
 {
     long i;
     for (i = 0; i < 1000000; i++) {}
 }

void CALCULOS()
{
               VVABRMS= PVABRMS*100;   VVBCRMS=PVBCRMS*100;
VVCARMS=PVCARMS*100;
                VIARMS =IARMS*100;   VIBRMS=IBRMS*100;   VICRMS=ICRMS*100;
VINRMS=INRMS*100;

                VFrec=Frec*100; VFrec2=Frec2*100; VFrec3=Frec3*100;
                Vfase2=fase2*100;Vfase3=fase3*100;
                Vdesfase24=desfase24*100; Vdesfase25=desfase25*100;
Vdesfase26=desfase26*100;


}
```

```
void secuenciaenviar(){

    tipo=1; condiok1=65;
    divideval(VVABRMS);  //VR-S
    tipo=2; condiok1=66;
    divideval(VVBCRMS);  //VS_T
    tipo=3; condiok1=67;
    divideval(VVCARMS); //VT-R
    tipo=4; condiok1=68;
    divideval(VIARMS);  //IR
    tipo=5; condiok1=69;
    divideval(VIBRMS); //IS
    tipo=6; condiok1=70;
    divideval(VICRMS); //IT
    tipo=7; condiok1=71;
    divideval(VINRMS); //IN
    tipo=8; condiok1=72;
    divideval(Vfase2); //fase2
    tipo=9; condiok1=73;
    divideval(Vfase3); //fase3
    tipo=10; condiok1=74;
    divideval(VFrec); // frec1
    tipo=11; condiok1=75;
    divideval(VFrec2); //frec2
    tipo=12; condiok1=76;
    divideval(VFrec3); //frec3
    tipo=13; condiok1=77;
    divideval(Vdesfase24); ///desf1
    tipo=14; condiok1=78;
    divideval(Vdesfase25); //desf2
    tipo=15; condiok1=79;
    divideval(Vdesfase26); //def3

    tipo=25; condiok1=80;
    divideval(fin);
    inicializacion();
}


 void tiempo(void){

       if (seg==59)
           {
           CpuTimer0.InterruptCount=0;
           min++;
           if (min==59)
                 {
                 min=0;
                 hora++;

                 if (hora==24)
                                       {
                                        hora=0;
                                        dia++;
```

```
                                                        if (dia==30 &&
mes==2)
                                                            {
                                                                 dia=1;
                                                                 mes++;
                                                            }

                                                         if (dia==31 &&
(mes==4 || mes==6 || mes==9 || mes==11))
                                                            {
                                                                   dia=1;
                                                                   mes++;
                                                            }

                                                         if (dia==32 &&
(mes==1 || mes==3 || mes==5 || mes==7 || mes==8 || mes==10 ))
                                                                {
                                                                     dia=1;
                                                                     mes++;

                                                                }
                                                         if ( dia== 32 &&
mes==12)
                                                                {
                                                                   mes=1;
                                                                   ano++;
                                                                   dia=1;
                                                                }

                                            }

                    }
           }

 }
```