

DESARROLLO DE MÓDULOS PARA SISTEMA DE MEDICIÓN BASADO EN UN AMPLIFICADOR SINCRÓNICO DIGITAL A FASE (LIA) IMPLEMENTADO EN UNA FPGA.

Zarate García Jazmín Araceli
ESCOM-IPN
Ingeniería en Sistemas
Computacionales
Av. Juan de Dios Batiz
esq.Othon de mendizabal
jazzy.sun220@gmail.com

Puebla Lomas Jaime Hugo
ESCOM-IPN
Ingeniería en Sistemas
Computacionales
Av. Juan de Dios Batiz
esq.Othon de mendizabal
jpuebla@ipn.mx

Carcaño Miranda Silvia Edith
ESCOM-IPN
Ingeniería en Sistemas
Computacionales
Av. Juan de Dios Batiz esq.
Othon de mendizabal
decamij@gmail.com

RESUMEN.

Recientemente se demostró la posibilidad de implementar un amplificador sincrónico digital sensible a fase (LIA) ^{[1],[2],[3]},

utilizando un arreglo de compuertas programable en campo (FPGA) ^[4]. Es por ello y debido a las necesidades de las técnicas foto térmicas, las cuales se basan en el estudio de las variaciones de la transferencia de calor sobre diversos materiales, las cuales son medibles con dispositivos especializados en detección de amplitud y fase, se desarrolla este trabajo, complementando la programación del arreglo de compuertas programables en campo, además de implementar un sistema de mediciones intuitivo al usuario final.

Palabras Clave: Electrónica analógica, FPGA, Interface, Sistema y Procesamiento Digital de Señales.

ABSTRACT.

The possibility of implementing a phase sensitive digital synchronous amplifier (LIA) ^{[1],[2],[3]} using a field programmable gate array (FPGA) ^[4] was recently demonstrated.

It is for this reason and due to the needs of photothermal techniques, which are based on the study of the variations of heat transfer on various materials, which are measurable with devices specializing in amplitude and phase detection, this work is developed, complementing the programming of the arrangement of programmable gates in the field, in addition to implementing an intuitive measurement system to the end user.

Keywords: Analog Electronics, FPGA, Interface, System and Digital Signal Processing.

1. INTRODUCCIÓN

Las técnicas fototérmicas, permiten analizar diversos fenómenos de transporte de calor, y masa en materiales sólidos, líquidos y gases. El análisis de este tipo de fenómenos requiere de operaciones complejas, lo que involucra el uso de

múltiples circuitos para el análisis y proceso de las señales medidas. Para llevar a cabo el análisis de la respuesta del material de estudio, es común utilizar un amplificador sincrónico o Lock-In Amplifier (LIA).

Los LIA son capaces de medir señales periódicas, de voltaje o de corriente, de amplitudes muy pequeñas (del orden de micros y nano voltios) y una alta selectividad de frecuencia permitiendo distinguir entre la señal de interés y el ruido de fondo ^{[1],[5]}.

Los LIA tienen la característica de ser demasiados costosos, voluminosos en cuanto al espacio que ocupan, y no siempre se utilizan todas sus funciones, lo cual provoca que el trabajo sea lento, ya que se pierde tiempo en adaptar el LIA al experimento.

Actualmente, existen propuestas de prototipos más portables que cumplen con dicha función y que reducen los problemas anteriormente mencionados, aunque algunos tienen un costo elevado y no son accesibles para los usuarios.

En CICATA – IPN, Legaría se desarrolló un prototipo de LIA mucho más portable y accesible al usuario, dicho dispositivo es completamente funcional, en comparación con los LIA que existen, actualmente en el mercado u otros dispositivos similares, también dicho prototipo presenta una serie de áreas de oportunidad a optimizar.

El trabajo que se describirá a lo largo de este artículo, se centrará en abordar las áreas de oportunidad encontradas en el prototipo existente en CICATA–IPN unidad Legaría, una buena optimización de estos módulos nos ayudará a obtener un mejor dispositivo para sus laboratorios y atraer alumnos politécnicos que desarrollen este tipo de tecnologías no solo para el IPN sino para nuestro país.

2. METODOLOGIA.

Revisando diversas metodologías se propuso desarrollar nuestro trabajo terminal utilizando la metodología o modelo de desarrollo por Prototipado debido a las características mencionadas en los siguientes puntos.

2.1 Modelo de Desarrollo por prototipado.

Este modelo recomienda el desarrollo de una implementación inicial del sistema, que debe ser presentada al cliente para su negociación, y posteriormente refinada mediante versiones hasta alcanzar el desarrollo completo del sistema.

Este modelo de desarrollo también es conocido como evolutivo, en donde las actividades son llevadas a cabo en forma concurrente y tienen retroalimentación en todo el proceso [10].

De este modelo de desarrollo se detectan varios enfoques:

- Modelo incremental
- Modelo en espiral
- El desarrollo basado en componentes
- Desarrollo por prototipos

Según las necesidades detectadas a lo largo del pre análisis de nuestro proyecto se implementará el desarrollo por prototipos.

Desarrollo de Prototipos: El modelo de desarrollo por prototipos se utiliza cuando los usuarios no tienen claros sus requerimientos al inicio del sistema. Un prototipo es una versión inicial de un sistema que se utiliza para demostrar conceptos, opciones de diseño y en general propuestas de solución [9].

Existen dos tipos de desarrollo por prototipos:

- Desarrollo Exploratorio
- Prototipos Desechables

El modelo de desarrollo por prototipos del tipo exploratorio; fue la metodología que se implementó a lo largo de este desarrollo ya que en este tipo de desarrollo se trabaja con el cliente para encontrar sus requerimientos y entregar un sistema final. El sistema es mejorado cada vez que existan nuevas propuestas del cliente [10],[11].

Lo que nos permitió añadir nuevos elementos sobre el prototipo inicial, para refinar el desarrollo según los comentarios del cliente o usuario final, con los cuales se podrán modificar o hacer reingeniería con cada iteración que realicemos a cada

etapa del proyecto con el fin de mejorar el desarrollo hasta llegar a un prototipo final usable por el usuario final.

El modelo de desarrollo por prototipos se compone de una serie de pasos:

- Recolección de requisitos
- Diseño rápido
- Prototipo
- Nueva iteración

2.2 Modelo de desarrollo por prototipos propuesto.

El prototipo propuesto que conforma el LIA, para este trabajo terminal, se compondrá de tres módulos. Cada módulo del prototipo, proporcionará una función esencial para el funcionamiento del LIA.

Los módulos que conforman el prototipo propuesto son los siguientes:

- Módulo Filtro: El cual se encargará de acondicionar la señal de entrada, la cual se entregará lista para ser utilizada por el módulo de la FPGA.
- Módulo FPGA: Recibirá la señal acondicionada por el módulo del filtro y esperará instrucciones del módulo de la interfaz, para poner en funcionamiento sus tareas específicas.
- Módulo Interfaz: Se encargará de reconocer el módulo de la FPGA, por medio de un puerto de comunicación y enviará los datos necesarios para que el módulo de la FPGA comience su funcionamiento, para posteriormente mostrar los resultados obtenidos.

Por lo que la arquitectura general de nuestro sistema quedará descrita de la forma como lo muestra la figura 1.



Figura 1. Diagrama a bloques del Sistema

3. IMPLEMENTACIÓN MODULO FILTRO

Para el modulo del filtro, se diseñó un filtro Butterworth de -60dB, se escogió este tipo de filtro porque su banda de paso, recibo una señal más plana, después de su frecuencia de corte, esta comienza a decaer en -3 dB.

Este tipo de filtro, fue un filtro Pasa Bajas, ya que las frecuencias que se están midiendo son menores a los 15 Khz, aunque con respecto a la frecuencia de la FPGA, esta se estableció a los 6Khz.

Además, la estructura que se utilizó para este filtro fue Sallen-Key, se escogió este tipo de estructura por su simplicidad y flexibilidad.

Este tipo de estructura permite que su diseño sea cómodo, y que si se requiere más etapas estas se conecten en cascada.

Aparte del filtro, se diseñó una etapa de OFFSET, la cual evita, que se reciban voltajes negativos, los cuales puedan llegar a ser perjudiciales para la FPGA.

Además, la conexión de la FPGA se realiza por medio de un seguidor de voltaje, para que el voltaje no se atenuara en el circuito.

3.1 Diseño del Filtro.

En la figura 2, se muestra el diseño del filtro que se utilizo para la obtención de la frecuencia de corte para sincronizarlo con la frecuencia del LIA.

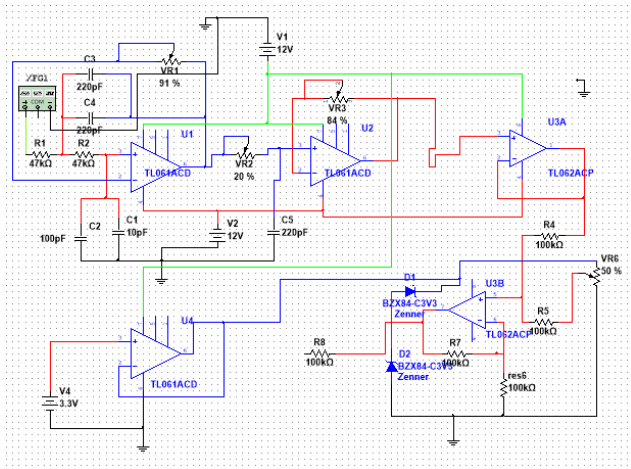


Figura 2. Esquematico del circuito representativo del filtro.

Después de realizar las pruebas, se construyo el PCB para que se disminuyera el ruido externo, obteniendo el circuito mostrado en la figura 3.

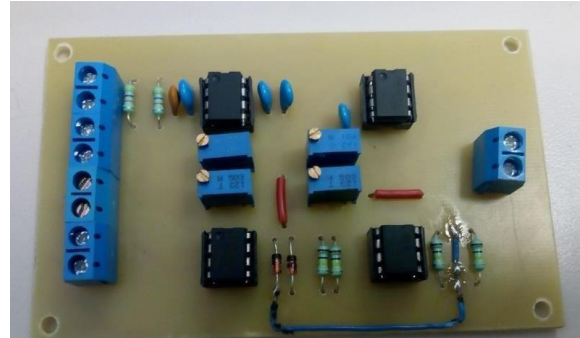


Figura 3. Placa del circuito de filtrado.

4. Modulo de Interfaz Gráfica.

El desarrollo de nuestro modulo la interfaz gráfica, se desarrolló en el lenguaje de programación JAVA, con ayuda del IDE NetBeans. Ya que como objetivos principales de este desarrollo es ofrecerle una interfaz de uso libre a nuestro usuario final, por ello del desarrollo en dicha plataforma.

Las pantallas principales de nuestro desarrollo quedaron descritas por las figuras 4 y 5.

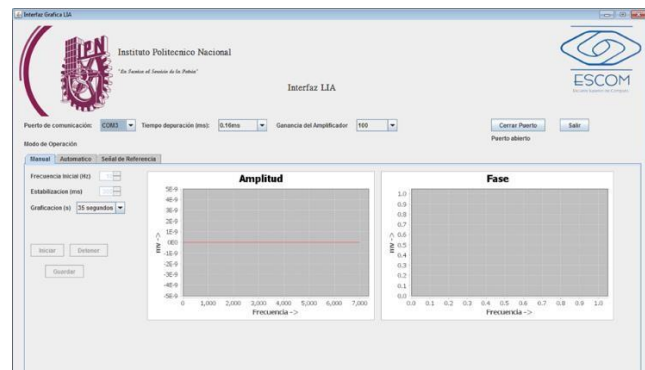


Figura 4. Interfaz manual del Sistema

El modo de operación manual se puede ver en la imagen anterior, para el modo de operación automático se llenan los campos que describe la imagen a continuación.

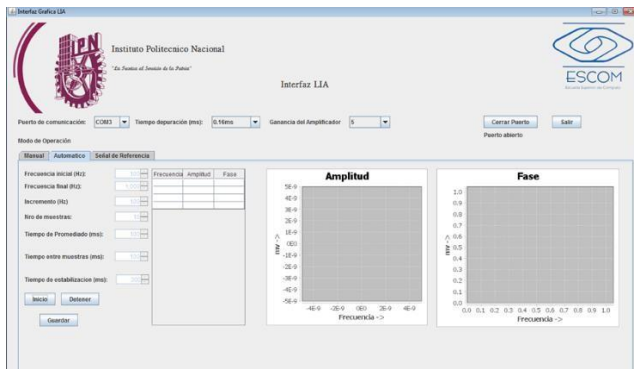


Figura 5. Interfaz automática del sistema.

Se tiene un modo de operación extra, el cual nos muestra las opciones de configuración con los que cuenta nuestro módulo FPGA, lo que nos permite visualizar estas opciones desde nuestra interfaz gráfica.

En la figura 6 se muestra la interfaz para graficar tanto el espectro de amplitud como el de fase para poder observar el comportamiento del sistema.



Figura 6. Interfaz espectro de amplitud y fase.

Para tener una óptima comunicación entre nuestros módulos FPGA e interfaz gráfica JAVA nos ofreció la facilidad de establecer la comunicación con ayuda de Giovynet Driver ya que faculta a Java, para interfazar circuitos electrónicos y/o circuitos electromecánicos desde un ordenador.

Lo que, en consecuencia, convierte a JAVA en una opción para el fabricante de hardware que desea comunicar sus creaciones con un PC. Giovynet Driver soporta el envío y recepción de datos a través del puerto serie, mediante una serie de métodos de la clase `giovynet.serial.Com`.

Para llevar a cabo el envío y recepción de datos utilizamos los métodos que nos ofrece Giovynet, en el caso del envío fue:

- `sendSingleData(overloaded)` : void Este método es usado para enviar un elemento de tipo char, o de tipo

String o de tipo Hex. El elemento se enviará transcurrido el tiempo determinado en el método `setMinDelayWrite( int milisegundos)` de la clase `giovynet.serial.Parameters`, por "default" el tiempo establecido para Windows es 0 milisegundos, y para Linux es de 10 milisegundos [5],[3].

Para la recepción de datos utilizamos el siguiente método:

- Método `receiveSingleCharAsInteger()` : int Este método es usado para recibir un dato ASCII en representación entera, luego del tiempo establecido por el método `setMinDelayWrite(int milisegundos)`, de la clase `giovynet.serial.Parameters`, por "default" el tiempo establecido para Windows es 0 milisegundos, y para Linux es de 10 milisegundos. Este método es recomendado para recibir caracteres de control tales como, <ACK>, <NACK>, <LF>, ...etc.

5. Implementación de la comunicación serial en la interfaz gráfica JAVA.

Para establecer una comunicación entre nuestro módulo de la FPGA e Interfaz Gráfica se implementó la clase llamada `FPGAConnection` la cual nos permite visualizar los puertos disponibles en nuestra PC. Primero se instancia un objeto de tipo `giovynet.nativelink.SerialPort`, seguidamente se utiliza el método `getFreeSerialPort()` para obtener una lista String de puertos libres:

```
public void FPGAConnection() {
    try {
        SerialPort puerto = new SerialPort();
        List<String> listaPuertos = puerto.getFreeSerialPort();
        //Tiempo de estabilización de la señal
        List<String> listatiempo = null;

        if (listaPuertos.isEmpty()) {
            throw new Exception("Lista Vacía");
        }

        for (int x = 0; x < listaPuertos.size(); x++) {
            ElijoP.addItem(listaPuertos.get(x));
        }
    } catch (Exception ex) {
        System.err.println("Error no se encontro ningun puerto di");
        ElijoP.addItem("Vacío");
        abrirpuerto.setEnabled(false);
        Finalizar.setEnabled(true);
    }
}
```

Para la apertura del puerto serial seleccionado se implementó el botón Abrir Puerto en el cual se establece la configuración del puerto seleccionado. Para configurar el puerto serie primero se crea un objeto de tipo `giovynet.serial.Parameter`, ya que cambiamos los parámetros por "default", se utilizó los métodos

set del objeto parameter, antes de instanciar la clase giovynet.serial.Com [5],[3].

De igual forma se utilizan los métodos sedSingleData y RecibeSingleData los cuales permiten abrir o cerrar el puerto de comunicación con la tarjeta enviado los comandos en ASCII que establecen la comunicación con la FPGA. Según lo establecido en los Módulos de la FPGA, los comandos que se enviaran para comenzar la Operación de la FPGA o cambio de configuración son los siguientes descritos en la tabla [1] :

Comando ASCII	Descripción
P	Detener Transmisión
S	Comenzar transmisión continua
E	Habilita oscilador interno
D	Deshabilita Oscilador interno
*	Cambie la frecuencia de operación, debe ir seguido de un número de 32 bits que es proporcional a la Frecuencia de operación.
#	Cambiar la opción de configuración, debe ir seguido de un número de 8 bits.

Tabla 1. Comandos de la FPGA

5.1. Descripción de comandos ASCII para control de la FPGA.

Entonces la función de nuestro botón Abrir Puerto mediante los métodos sedSingleData y RecibeSingleData envía los caracteres de control S y E respectivamente lo que nos permite realizar las tareas que se describieron en la tabla.

6. Conclusiones.

En la parte del filtrado, se logró realizar adecuadamente esta etapa de filtrado y acondicionamiento de la señal para que esta se reciba en la FPGA, y no cause ningún conflicto, ya que, al ser una tarjeta delicada, esta tiene que tener ciertos requerimientos, para no causar ningún daño. Ya con esta etapa realizada, se realizó el diseño y después la impresión del circuito en PCB, para la disminución de algún otro ruido externo, además, se estableció la comunicación entre la FPGA

y la interfaz gráfica por medio del puerto serial, lo cual, permitió que se enviara y recibieran los datos adecuadamente Por parte de la interfaz gráfica, esta misma esta adecuada para el estudio de las técnicas foto térmicas, y que el usuario, fuera más fácil de utilizarla, sin la necesidad que se cuente con el desarrollador o con alguna otra persona. Además, se añadió una función extra a la interfaz gráfica, la cual no estaba en los objetivos, por solicitud del personal del laboratorio, esta función, permite que se muestre cual es la señal que se está recibiendo de la FPGA.

7. Referencias.

- [1] C. A., "Implementación de un amplificador sincrónico digital (Lock-In) usando un dispositivo FPGA (Field Programmable Gate Array)" Tesis de Maestría, CICATA, Legaria, -IPN, México, 2013.
- [2] A. Cifuentes y E. Marín, Implementation of a Field Programmable Gate Array-based Lock-in Amplifier. Measurement, México, 2015. M.E. y R. Ivanov, E. Marín y R. Ivanov "LIA in a Nut Shell: How can trigonometry help to understand Lock- In Amplifier Operation", México: Latin American Journal of Physics Education, 2009.
- [3] «Xilinx Inc., Spartan-3E FPGA Family Data Sheet,» 2012. [En línea]. Available: http://www.xilinx.com/support/documentation/data_sheets/ds312.pdf.
- [4] B. G., "Unidad en Arquitectura FPGA modular y acoplable a sistemas científicos y tecnológicos que requieren análisis de fase y amplitud", Tesis de Doctorado, Universidad Autónoma de Querétaro, Facultad de Ingeniería, Querétaro, México,
- [5] R. R, Abbiati y A. Geraci , Abbiati and A. Geraci, "Digital field programmable gate array-based lock-in amplifier for high-performance photon counting applications", Rev. Sci. Inst 76, 2005.