ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad De Ingeniería En Electricidad Y Computación

Red de Sensores usando Plataforma NIOS II

TESINA DE SEMINARIO

Previa la obtención del Título de:

INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

Presentado por:

Lissette Viviana Aguilar Oliveros

Luis Adolfo Moreira Neira

GUAYAQUIL – ECUADOR

AÑO 2014

AGRADECIMIENTO

En primer lugar quisiera agradecer a Dios, porque sin él nada fuera posible.

Quiero agradecer a toda mi familia, en especial a mis padres que han sido testigos de cada uno de mis logros y el pilar fundamental para seguir siempre adelante sin darme por vencida, sin ellos no hubiera podido llegar a cumplir este objetivo.

A todos mis compañeros y sobre todo a mis amigos que formaron parte de mi vida universitaria, contribuyendo día a día a que pueda llegar a este día tan importante.

Lissette Viviana Aguilar Oliveros

Gracias a mis padres, que siempre han sido y son ese pilar fundamental en mi vida, sin sus sabios consejos, apoyo y orientación no sería la persona que soy ahora.

Luis Adolfo Moreira Neira.

DEDICATORIA

A mi familia y amigos, de manera especial a mis padres ya que sin ellos no hubiera podido lograr con éxito todo lo que soy ahora.

Lissette Viviana Aguilar Oliveros

A mis padres y amigos por toda su ayuda brindada, ya que sin ellos nada de lo obtenido podría haber sido posible.

Luis Adolfo Moreira Neira

TRIBUNAL DE SUSTENTACIÓN

Ing. Ronald Ponguillo

PROFESOR DEL SEMINARIO DE GRADUACIÓN

Ing. Víctor Asanza

PROFESOR DELEGADO POR LA UNIDAD ACADÉMICA

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de esta Tesina, nos corresponde exclusivamente; y el patrimonio intelectual del mismo a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL". (Reglamento de exámenes y títulos profesionales de la ESPOL)

Lissette Viviana Aguilar Oliveros

Luis Adolfo Moreira Neira

RESUMEN

El presente proyecto consiste en la implementación de una red de sensores inalámbricos aplicada en un sistema de riego para un invernadero utilizando la tarjeta de desarrollo **DE2-115 ALTERA,** basada en un dispositivo **FPGA CYCLONE IV** de Altera junto con memorias embebidas y un microprocesador **NIOS II** integrado que sirve para monitorear cultivos de un invernadero gracias a la obtención de datos de los distintos nodos sensores que se comunican a través del protocolo RS232 con la tarjeta DE2-115, que será el nodo central del sistema y el encargado de llevar una base de datos de todos los procesos que ejecuten cada uno de los nodos sensores almacenándolos en una memoria SD-CARD mediante una serie de algoritmos implementados en lenguaje C.

Para la realización del proyecto se implementó una topología de redes tipo malla, en donde cada uno de los dispositivos que actúan como nodos tienen comunicación con el resto de los nodos que conforman la malla y son capaces de efectuar mediciones en tiempo real, enviar los datos a través de módulos inalámbricos y receptarlos en la tarjeta DE2-115 quien será el nodo central

para ser mostrados en una pantalla LCD integrada en la misma y en base a esto tomar decisiones a ejecutar.

El proyecto ha sido divido en 5 capítulos como se detalla a continuación:

En el **primer capítulo**, se exponen los objetivos generales y específicos del proyecto, la identificación del problema y la metodología que usamos para la solución del mismo, así como también se indican los alcances y limitaciones del proyecto.

En el **segundo capítulo**, se indican los conceptos básicos del Hardware y Software utilizados durante el desarrollo del proyecto, adicionalmente se brindan breves conocimientos adquiridos con respecto a la agricultura.

En el **tercer capítulo**, se muestra el desarrollo del proyecto, presentando el diseño e implementación realizada paso a paso de las diferentes etapas que se ejecutan para implementar la solución de dicho proyecto.

Para el **cuarto capítulo**, se presentan los distintos casos y situaciones en las que se realizaron las respectivas pruebas para poder comprobar la efectividad y funcionamiento del proyecto.

Finalmente se generan las conclusiones sobre toda la funcionalidad del proyecto, lo cual nos permite generar recomendaciones que servirán como base para futuras aplicaciones o mejoras que se le quieran dar al presente proyecto.

ÍNDICE GENERAL

ABREVIATURAS

DE2	Development and Education Board
IDE	Integrated Development Environment
ISM	Industrial, Scientific and Medical
FPGA	Field Programmable Gate Array
RISC	Reduced Instruction Size Code
HDL	Hardware Description Language
RS-232	Recommended Standard 232
LDR	Ligth Dependent resistor
API	Application Programming Interface
AT	Advanced Technology
ZC	ZigBee Coordinator
ZR	ZigBee Router
ZED	ZigBee End Device

FFD	Full Functionality Device
RFD	Reduced Functionality Device
ADC	Analog to Digital Converter
CPU	Central Processing Unit
EUSART	Enhanced Universal Synchronous/Asynchronous Receiver/ Transmitter
EEPROM	Electrically Erasable Programmable Read Only Memory
HW	Hardware
JTAG	Joint Test Action Group
SD CARD	Secure Digital Card
LCD	Liquid Cristal Display
PS/2	Personal System / 2
IRDA	Infrared Data Association
ASCII	American Estándar Code for Information Interchange
HAL	Hardware Abstraction Layer

RAM	Random Access Memory
ROM	Read Only Memory
ASIC	Application Specific Integrated Circuit
DSP	Digital Signal Processor
BOR	Brown Out Reset
PWM	Pulse Width Modulation
USART	Universal Synchronous Asynchronous Receiver Transmitter
MSSP	Master Serial Sinchronous Port
SPI	Serial Peripheral Interface
NZR	Non Return to Zero
IEEE	Institute of Electrical and Electronic Engineers
VHDL	VHSI Hardware Description Language
JDT	Java Development Toolkit
SWT	Standar Widget Toolkit

ISIS	Intellingent Schematic Input System
VMS	Virtual System Modelling
ARES	Advanced Routing Modelling
PCB	Printed Circuit Board
LV-TTL	Low Voltage Transistor Transistor Logic
TTL	Transistor Transistor Logic
TX	Transmisión
RX	Recepción

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad la tecnología ha avanzado de una manera impresionante y eficaz, siempre buscando métodos más sencillos, robustos, ágiles y que proporcionen una mayor interacción con el usuario.

Los sistemas basados en **FPGA** son aquellos capaces de cumplir con dichos requisitos, ya que pueden ser programadas por el usuario de una manera tal que cumplan con especificaciones y/o requerimientos definidos por los mismos y no verse limitados a soluciones que brindan los fabricantes. La única restricción para estos dispositivos es la creatividad e ingenio para desarrollar soluciones de los usuarios, así como también pueden ser complementados a otros sistemas otorgando mayor eficiencia, alcance y complejidad de una manera personalizada de acuerdo a sus necesidades.

Actualmente la comunicación inalámbrica forma parte importante en todo diseño de ingeniería por los múltiples beneficios que ofrece, brindando alta confiabilidad al momento de la transmisión y recepción de datos, como lo es el caso de la tecnología Zigbee.

Al juntar la tecnología Zigbee con los FPGA's se pueden encontrar diversas soluciones a problemas tales como la seguridad, comunicación, monitoreo continuo de procesos y muchas otras necesidades de la vida cotidiana.

Uno de los procesos que deben ser monitoreados de manera continua es el cultivo en invernaderos, ya que los cultivos deben ser regados constantemente de acuerdo a la temperatura, humedad u horario de riego.

En ciudades como Guayaquil donde los factores climatológicos varían frecuentemente es necesario tener un cuidado continuo de los cultivos o en su defecto tener dispositivos dedicados al monitoreo de la producción de los mismos. En el mercado existen dispositivos encargados de controlar los procesos que se realizan en los invernaderos, muchos de estos dependen de un nodo central encargado del control de todas las señales y la toma de decisiones, pero no dan la facilidad de poder programarlos para los diferentes tipos de cultivos que desee cultivar el usuario. Nuestra red de sensores ofrece

la facilidad al usuario de poder ingresar el tipo de cultivo y las especificaciones del mismo para que de esta manera el sistema sea capaz de tomar decisiones de manera diferenciada con respecto a otros cultivos.

El siguiente proyecto presenta un sistema de sensores inalámbricos usando la tarjeta de desarrollo DE2-115 de ALTERA para el monitoreo y control de un invernadero de manera inteligente comunicado por módulos Xbee, en donde el usuario puede ingresar los criterios de cuidado para cada cultivo.

CAPÍTULO 1

1. GENERALIDADES

En este capítulo se muestra la identificación del problema, se presenta el planteamiento del proyecto, sus objetivos, alcances y limitaciones, así como también la metodología con la que se va a resolver el mismo.

1.1. OBJETIVOS

1.1.1. OBJETIVOS GENERALES

- ✓ Construir un sistema embebido basado en el microprocesador NIOS II y bloques de lógica reconfigurable que permitan controlar una red de sensores inalámbricos que puedan conectarse y colaborar entre

sí en la solución de problemas específicos que presente nuestra sociedad.

- ✓ Diseñar e implementar un prototipo de red inalámbrica tipo malla que sea segura, confiable, fácil de usar, de bajo consumo de energía y de bajo costo, para el monitoreo y control de riegos en una amplia gama de sectores y cultivos de un invernadero.

1.1.2. OBJETIVOS ESPECÍFICOS

- ✓ Obtener conocimientos acerca del funcionamiento de los FPGA, con la finalidad de utilizarlo como controlador del sistema de red de sensores.
- ✓ Aprender el funcionamiento y aplicación de la tarjeta de desarrollo DE2-115 de ALTERA.
- ✓ Generar un respaldo de todas las actividades realizadas en el invernadero durante el día mediante la implementación de una solución HW/SW y almacenarlas en una base de datos dentro de una SD Card.
- ✓ Diseñar e implementar un sistema de sensores inalámbricos para el control de un sistema de riego inteligente para un invernadero.

1.2. ALCANCE Y LIMITACIONES

La tarjeta de desarrollo DE2-115 de Altera nos brinda algunas alternativas para poder cumplir con los requisitos básicos de nuestra red de sensores, pero siendo este un proyecto con finalidad académica se presentan a continuación alcances y limitaciones del mismo:

ALCANCE

- ✓ El sistema permite el ingreso de características específicas dependiendo del tipo de cultivo que se desea monitorear tales como: humedad y temperatura.
- ✓ La información obtenida por los sensores será almacenada en una SD Card en formato .xls para una mejor interpretación por el usuario y a partir de estos datos se podrá generar un gráfico de Humedad vs. Temperatura para poder observar la correlación entre estos dos parámetros.
- ✓ El alcance de comunicación por medio de los módulos Xbee Serie 2 es de 120 metros con línea de vista y de 40 metros con obstáculos.
- ✓ El sistema de riego utilizado es mediante goteo.

LIMITACIONES

- ✓ Puesto que nuestro sistema está implementado con una red tipo malla no posee limitaciones en el área de comunicación y toma de decisiones, ya que si uno de los nodos se cae siempre se mantiene la

alta disponibilidad entre el resto de nodos para poder tomar así una resolución.

1.3. IDENTIFICACIÓN DEL PROBLEMA

En cualquier campo de la ingeniería existe la necesidad de llevar un registro de varios parámetros de importancia en cierta aplicación, el cual permite a las personas tener información importante para realizar análisis que ayuden a prevenir posibles errores presentes en dicha aplicación.

Conforme a esto hemos utilizado dispositivos con capacidad de llevar un registro de diferentes tipos de parámetros en tiempo real. Para comodidad de los usuarios es necesario que estos dispositivos sean lo más pequeños posibles permitiendo la portabilidad además de tener una interfaz amigable, interactiva e intuitiva.

En la actualidad el uso de las FPGA's es una alternativa que está siendo implementada por los desarrolladores de hardware y software para optimizar el uso de recursos (tiempo de procesamiento y sobre todo dinero).

Teniendo en cuenta lo mencionado anteriormente decidimos implementar una red inalámbrica de sensores, la cual tiene como núcleo principal el microprocesador NIOS II que será embebido en una FPGA CYCLON IV de

la compañía Altera y para la transmisión inalámbrica utilizamos módulos Xbee Serie 2 puesto que manejaremos una topología de red tipo malla.

Los nodos se comunican entre ellos con un ordenador o en este caso con una Tarjeta de Desarrollo DE2-115 de manera inalámbrica, transmitiendo periódicamente los datos recogidos. Aunque originalmente las redes inalámbricas de sensores se pensaron para nodos fijos, en la actualidad se han desarrollado sistemas con nodos móviles.

1.4. METODOLOGÍA

En nuestra red inalámbrica de sensores implementada para el sistema de riego de un invernadero cada nodo está encargado de medir las condiciones o factores ambientales para un cultivo específico de acuerdo a las necesidades del usuario. Los factores a medir son:

- ✓ Humedad del suelo.
- ✓ Temperatura Óptima de Cultivo.
- ✓ Porcentaje de Luminosidad.

Por ser una red tipo malla, todos los nodos están comunicados entre sí.

Para establecer la comunicación inalámbrica utilizamos la tecnología Zigbee tipo malla, lo cual nos evita el trabajo de hacer una infraestructura cableada adicional y exclusiva para el sistema, y sólo es necesario

seleccionar el lugar adecuado e instalarlos, de igual manera estos transmisores reducen el consumo de energía ya que solo requieren una batería de 9 Voltios.

En cuanto al almacenamiento de datos obtenidos por cada sensor hacemos uso del puerto de almacenamiento SD Card guardándolos en formato .xls para facilitar la interpretación del usuario conjuntamente con un gráfico a partir de la misma información de la tabla obtenida de los valores de humedad y temperatura.

1.5. HARDWARE Y SOFTWARE A UTILIZAR

HARDWARE

- ✓ Tarjeta DE2-115 – Altera.
- ✓ Sensor de Temperatura – DS18B20
- ✓ Sensor de Humedad - SKU: 3202591
- ✓ Sensor de Luminosidad - LDR
- ✓ Microcontrolador PIC 16F887
- ✓ Módulos Xbee Serie 2

SOFTWARE

- ✓ Quartus II 12.1
- ✓ NIOS II SBT 12.1
- ✓ Proteus 7.6
- ✓ MikroC 6.0
- ✓ X – CTU 5.2.7.5
- ✓ PicKit 2 V2.61

CAPÍTULO 2

2. MARCO TEÓRICO

En este capítulo se muestran conceptos básicos y funcionamientos de los programas y tecnologías que fueron utilizadas para el desarrollo del proyecto. Se presenta de manera básica el funcionamiento de la tarjeta de desarrollo DE2-115 de Altera, así como la función de cada uno de los módulos usados en el proyecto además se muestra las características principales del microcontrolador PIC 16F887 y de cada uno de los sensores utilizados.

2.1. TARJETA DE DESARROLLO DE2-115

2.1.1. CARACTERÍSTICAS

La Tarjeta DE2-115 de Altera consta de varias características que permiten al usuario practicar sus conocimientos mediante el desarrollo de diferentes circuitos digitales desde los más simples como un arreglo de puertas lógica, hasta los que demandan más uso de sus periféricos. Esta tarjeta cuenta con una FPGA CYCLONE IV para fines educativos y de investigación.

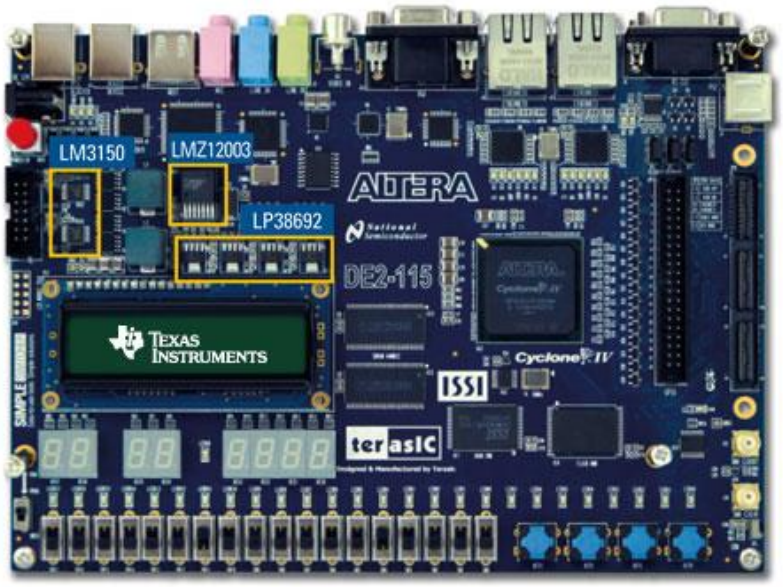


Figura 2.1 Tarjeta de Desarrollo DE2-115 de Altera [1]

A continuación se muestran las diferentes partes que conforman a la tarjeta DE2-115 de Altera:

FPGA:

- ✓ Altera Cyclone IV EPC4CE115.

Interfaces de Comunicación:

- ✓ USB Blaster para configurar la FPGA.

Memorias:

- ✓ 2–Mbyte SRAM.
- ✓ 128-Mbyte SDRAM.
- ✓ 8-Mbyte Memoria Flash.
- ✓ 32-Kbyte EEPROM.

Leds y Botones:

- ✓ 4 Pulsadores.
- ✓ 18 Switches.
- ✓ 18 Leds Rojos.
- ✓ 9 Leds Verdes.

Fuentes de Reloj:

- ✓ 3 Osciladores.

- ✓ 2 SMA conectores para fuente de reloj externo.

Audio y Video:

- ✓ Línea In/Out, Micrófono (24-bit Audio CODEC)
- ✓ VGA DAC de 10 bits.
- ✓ Decodificador de TV (NTSC/PAL)

Ethernet:

- ✓ 2 puertos 10/100/1000 Mbps controlador Ethernet con socket.

USB Host/ Esclavo:

- ✓ USB tipo A
- ✓ USB tipo B

Conectores:

- ✓ Conector para tarjetas SD.
- ✓ Interfaz RS 232 con conector DB9 de 9 pines.
- ✓ Conector PS/2 (ratón, teclado).
- ✓ 40 pines de expansión.

Varios:

- ✓ Módulo LCD de 16x2.
- ✓ 8 Displays de 7 segmentos.
- ✓ Transceptor Infrarrojo IrDA.

2.1.2. INTERFAZ LCD

La pantalla de cristal líquido o LCD es un dispositivo controlado de visualización gráfica para la presentación de caracteres, símbolos o incluso dibujos en alguno modelos.

El modelo de la tarjeta DE2-115 es de 2 filas de 16 caracteres donde cada una dispone de una matriz de 5x7 puntos.

Figura 2.2 Módulo LCD 16x2 [10]

Entre las características principales tenemos:

- ✓ Pantalla de caracteres ASCII.
- ✓ Desplazamiento de caracteres hacia la izquierda o derecha.

- ✓ Movimiento del cursor.

Para poder lograr la comunicación entre el microprocesador NIOS II y la pantalla LCD se la realiza a través de la Interfase Avalon-MM Slave.

El controlador LCD está acompañado de los archivos generados por el software, los cuales definen la interfaz de bajo nivel con el hardware y proporcionan los controladores HAL.

altera_up_avalon_character_lcd_regs.h

En este archivo se encuentran las declaraciones de funciones y constantes simbólicas para acceder al hardware de bajo nivel.

altera_up_avalon_character_lcd.c

Este archivo implementa las funciones necesarias para el manejo del dispositivo LCD, los cuales se generan a partir de la librería de sistema HAL.

2.1.3. PROTOCOLO RS-232

El protocolo RS-232 es una norma o estándar mundial que rige los parámetros de uno de los modos de comunicación serial. Por medio de este protocolo se estandarizan las velocidades de transferencia de datos, la forma de control que utiliza dicha transferencia, los niveles de voltajes utilizados, el tipo de cable permitido, las distancias entre

equipos, los conectores, etc. Además de las líneas de transmisión (Tx) y recepción (Rx), las comunicaciones seriales poseen otras líneas de control de flujo (Hand-shake), donde su uso es opcional dependiendo del dispositivo a conectar. A nivel de software, la configuración principal que se debe dar a una conexión a través de puertos seriales. RS-232 es básicamente la selección de la velocidad en baudios (1200, 2400, 4800, etc.), la verificación de datos o paridad (paridad par o paridad impar o sin paridad), los bits de parada luego de cada dato (1 ó 2), y la cantidad de bits por dato (7 ó 8), que se utiliza para cada símbolo o carácter enviado. La Norma RS-232 fue definida para conectar un ordenador a un modem. Además de transmitirse los datos de una forma serie asíncrona son necesarias una serie de señales adicionales, que se definen en la norma. Las tensiones empleadas están comprendidas entre +15/-15 voltios. [9]

2.1.4. MÓDULO SD CARD

La tarjeta de desarrollo DE2-115 cuenta con un puerto para la conexión de tarjetas SD, para poder utilizarlo necesitamos el módulo de control el cual es proporcionado por el programa Universitario de Altera que nos da acceso al Hardware necesario para el manejo de este dispositivo, el núcleo se ha diseñado para ser implementado mediante el generador Qsys.

2.2. FPGA

2.2.1. DEFINICIÓN

Una FPGA (Field Programable Gate Array) es un dispositivo semiconductor que contiene bloques de lógica cuya interconexión y funcionalidad puede ser configurada 'in situ' mediante un lenguaje de programación especializado. [1]

2.2.2. CARACTERÍSTICAS

Una de sus características importantes es que mediante la lógica programable dicho dispositivo es capaz de realizar desde las funciones más sencillas como las que son llevadas a cabo por medio de una puerta lógica o un sistema combinatorial, hasta complejos sistemas en un chip.

Las FPGA's poseen diferentes tecnologías de memoria: volátiles basadas en RAM y no volátiles basadas en ROM.

Los principales beneficios de los FPGAs son los siguientes:

- ✓ Flexibilidad y capacidad de desarrollo
- ✓ Precio
- ✓ Fiabilidad

- ✓ Reconfigurables

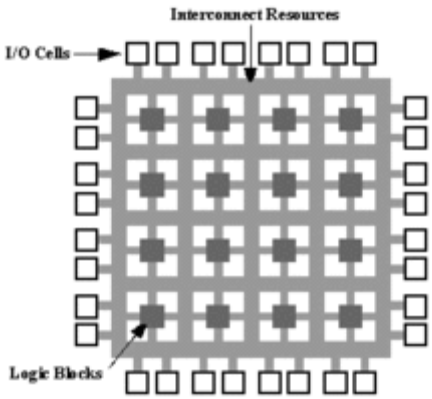


Figura 2.3 Arquitectura Básica de una FPGA [1]

2.2.3. APLICACIONES

Las FPGA's se utilizan en aplicaciones similares a los ASIC's sin embargo son más lentas, tienen un mayor consumo de potencia y no pueden abarcar sistemas tan complejos como ellos. A pesar de esto, las FPGA's tienen las ventajas de ser reprogramables (lo que añade una enorme flexibilidad al flujo de diseño), sus costos de desarrollo y adquisición son mucho menores para pequeñas cantidades de dispositivos. [4]

Hoy las FPGA's están presentes en campos tan diversos como la automoción, la electrónica de consumo, o la investigación espacial. La

tecnología FPGA tiene una aplicación horizontal en todas las industrias que requieren computación a alta velocidad. [5]

Tiene cabida en empresas dedicadas a la fabricación de sistemas electrónicos para: climatización de autobuses, comunicaciones por fibra óptica, conducción automática de trenes, control industrial, etc.

2.2.4. FPGA CYCLONE IV

La FPGA CYCLONE IV sirve para una amplia gama de aplicaciones lógicas generales, son ideales para aplicaciones de bajo costo y de factor de forma pequeño en los sectores de dispositivos inalámbricos y convencionales.

Figura 2.4 FPGA Cyclone IV [2]

Características:

- ✓ Elementos lógicos desde 6K a 150K [Ω].
- ✓ Memoria integrada de hasta 6.3 Mb.
- ✓ Posee hasta 360 multiplicadores de 18 x 18 para aplicaciones intensivas de procesamiento DSP.
- ✓ FPGA de bajo costo y baja potencia.
- ✓ Aplicaciones de conexión de protocolo para una alimentación total inferior a 1.5 W. [2]

2.3. MICROPROCESADOR EMBEBIDO NIOS II

Un sistema NIOS II es equivalente a un microcontrolador que incluye un procesador y una combinación de periféricos y memoria en un solo chip. Un sistema de un procesador NIOS II contiene: un núcleo procesador NIOS II, un conjunto de periféricos, memoria e interfaces, todas estas implementadas en un solo dispositivo de Altera. Como una familia de microcontrolador, todos los sistemas del procesador NIOS II usan un conjunto de instrucciones consistentes y un modelo de programación. [1]

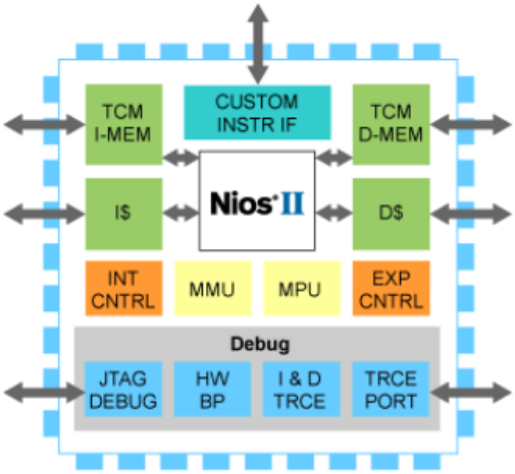


Figura 2.5 NIOS II System [3]

2.3.1. CARACTERÍSTICAS Y ARQUITECTURA

Entre sus características principales el NIOS II es un procesador configurable de 32 bits de propósito general, basado en una arquitectura tipo Harvard ya que usa buses separados para instrucciones y datos, utiliza un juego de instrucciones RISC, tiene una capacidad de direccionamiento de 32 bits, cuenta con acceso a variedad de periféricos integrados e interfaces para manejo de memorias y periféricos.

Existen 3 versiones disponibles en las que se puede implementar según se busque minimizar el consumo de recursos de la FPGA o maximizar el rendimiento del procesador:

- ✓ NIOS II /f (rápido): Es de alto rendimiento, con pipeline (conjunto de elementos procesadores conectados en serie) de 6 etapas aumenta su desempeño con opciones específicas como: memorias caché de instrucciones y datos o una unidad de manejo de memoria.

- ✓ NIOS II /s (estándar): Tiene un *pipeline* de 5 etapas que con una unidad aritmética lógica busca combinar rendimiento y consumo de recursos.

- ✓ NIOS II /e (económico): Requiere de menos recursos de la FPGA, no posee pipeline y es muy limitada porque carece de las operaciones de multiplicación y división.

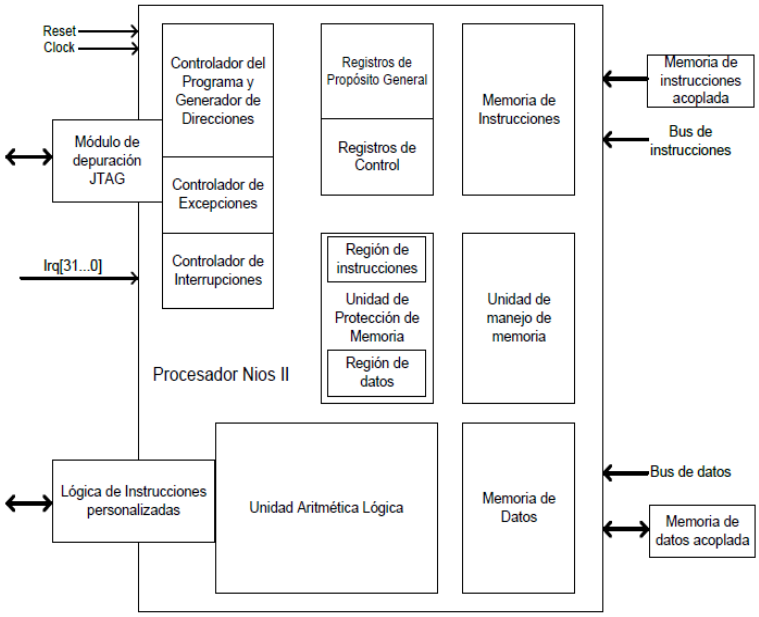


Figura 2.6 Estructura del Procesador NIOS II [4]

2.3.2. PROGRAMACIÓN DEL NIOS II

Para la programación del NIOS II se dispone de un entorno completo, basado en Eclipse, con el que Altera da soporte para el desarrollo de aplicaciones en un lenguaje C/C++. [3]

Debido a que utiliza este entorno se emplea una capa de software que oculta los detalles de la configuración del hardware, haciendo transparente al programador el desarrollo de aplicaciones.

2.4. MICROCONTORLADOR PIC 16F887

El microcontrolador PIC 16F887 es un dispositivo fabricado por la compañía Microchip, es uno de los más usados en la actualidad debido a su bajo precio, amplio rango de aplicaciones, alta calidad y disponibilidad, además es una solución perfecta para controlar cualquier tipo de proceso a nivel de empresas como para proyectos interactivos.

En la figura 2.7 se muestra un diagrama de bloques del microcontrolador PIC 16F887 donde se aprecian los módulos que integra y la conexión de cada uno.

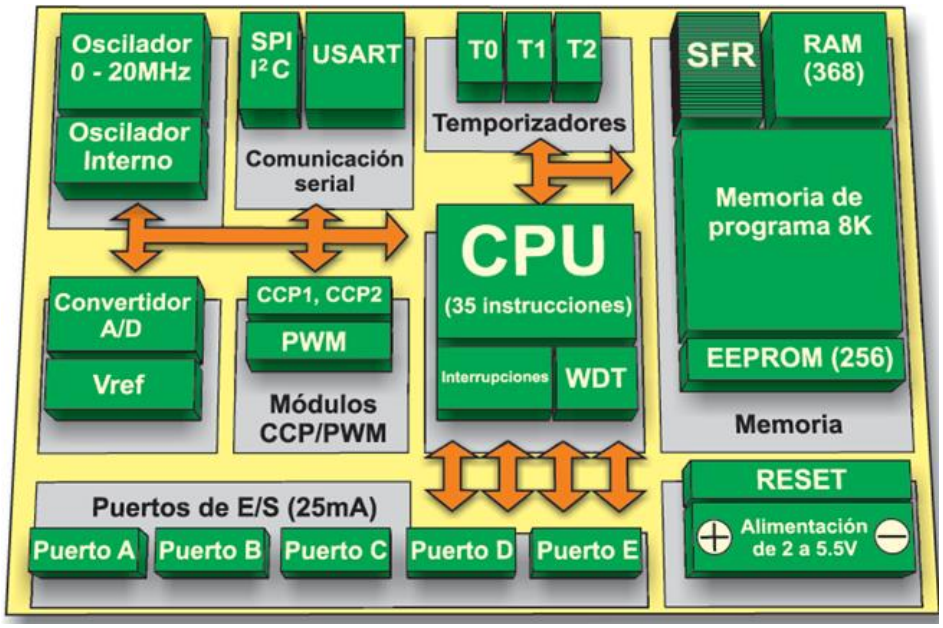


Figura 2.7 Diagrama de Bloques PIC 16F887 [11]

Entre las principales características tenemos:

- ✓ Arquitectura RISC.
- ✓ El microcontrolador cuenta con 35 instrucciones diferentes.
- ✓ Todas las instrucciones son uni-ciclo excepto por las de ramificación.
- ✓ Frecuencia de Operación 0 – 20 MHz.
- ✓ Oscilador interno de alta precisión.
- ✓ Rango de frecuencia desde 31KHz hasta 8MHz seleccionado por software.

- ✓ Voltaje de la fuente de alimentación de 2.0 – 5.5 V.
- ✓ Consumo: 220uA (2.0 V, 4MHz), 11uA (2.0 V, 32 KHz), 50 nA (en modo de espera.
- ✓ Ahorro de energía en el modo de suspensión.
- ✓ BOR con opción para controlar por software.
- ✓ 35 Pines de entrada/salida.
- ✓ Alta corriente de fuente y de drenado para manejo de LED.
- ✓ Resistencias pull-up programables individualmente por software.
- ✓ Interrupción al cambiar el estado del pin.
- ✓ Memoria ROM de 8K con tecnología FLASH.
- ✓ El chip se puede reprogramar hasta 100.000 veces.
- ✓ Opción de programación serial en el circuito.
- ✓ 256 Bytes de memoria EEPROM.
- ✓ Los datos se pueden grabar más de 1.000.000 veces.
- ✓ 368 Bytes de memoria RAM.

- ✓ Convertidor A/D:
  - 14 Canales.
  - Resolución de 10 bits.
- ✓ 3 Temporizadores/contadores independientes.
- ✓ Temporizador perro guardián.
- ✓ Modulo comparador analógico con:
  - Dos comparadores analógicos.
  - Referencia de voltaje fija (0.6V).
  - Referencia de voltaje programable en el chip.
- ✓ Módulo PWM incorporado.
- ✓ Módulo USART mejorado:
  - Soporta las comunicaciones seriales RS-485, RS-232 y LIN2.0.
  - Auto detección de baudios.
- ✓ MSSP:
  - Soporta los modos SPI e I2C.

2.4.1. MÓDULOS ANALÓGICOS

El módulo del convertidor A/D dispone de las siguientes características:

- ✓ El convertidor genera un resultado binario de 10 bits utilizando el método de aproximaciones sucesivas y almacena los resultados de conversión en los registros ADC (ADRESL y ADRESH);
- ✓ Dispone de 14 entradas analógicas separadas;
- ✓ El convertidor A/D convierte una señal de entrada analógica en un número binario de 10 bits;
- ✓ La resolución mínima o calidad de conversión se puede ajustar a diferentes necesidades al seleccionar voltajes de referencia Vref- y Vref+.

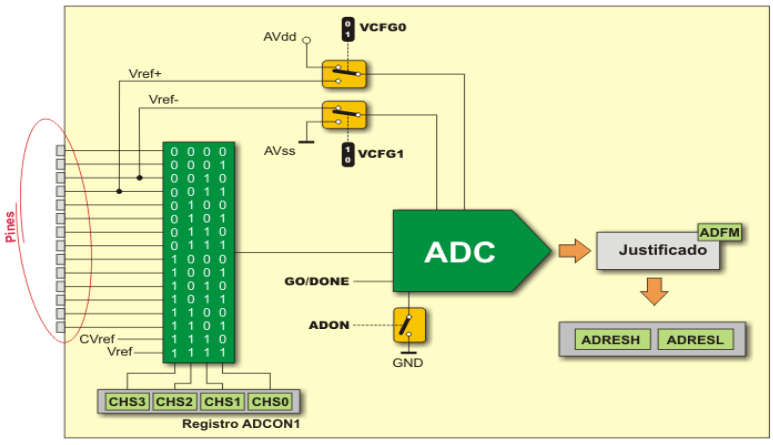


Figura 2.8 Módulo ADC PIC 16F887 [11]

El funcionamiento del convertidor A/D está bajo el control de los bits de cuatro registros:

- ✓ ADRESH Registro alto del resultado de la conversión A/D.
- ✓ ADRESL Registro bajo del resultado de la conversión A/D.
- ✓ ADCON0 Registro de control 0
- ✓ ADCON1 Registro de control 1

2.4.2. MÓDULO DE COMUNICACIÓN SERIE

El USART es uno de los primeros sistemas de comunicación serie. Las versiones nuevas de este sistema están actualizadas y se les denomina un poco diferente - EUSART.

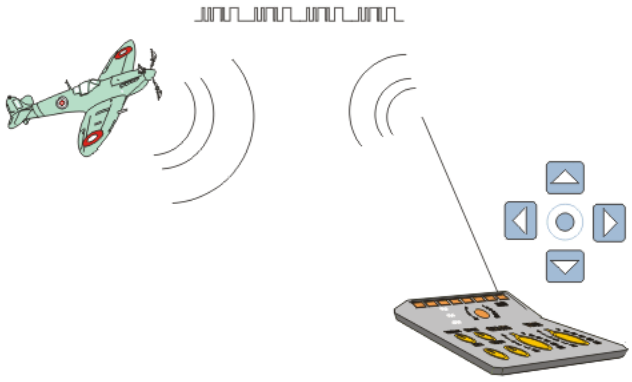


Figura 2.9 Comunicación entre dispositivos diferentes - EUSART [11]

El EUSART integrado en el PIC16F887 posee las siguientes características:

- ✓ Transmisión y recepción asíncrona en modo Full-Duplex.
- ✓ Caracteres de anchura de 8 – 9 bits programables.
- ✓ Detección de dirección en modo de 9 bits.
- ✓ Detección de errores por saturación del búfer de entrada.
- ✓ Comunicación Half Duplex en modo síncrono.

El EUSART en modo asíncrono transmite y recibe los datos utilizando la codificación de no retorno a cero – NRZ (non – return – to – zero), no se utiliza una señal de reloj y los datos se transmiten de forma muy simple:

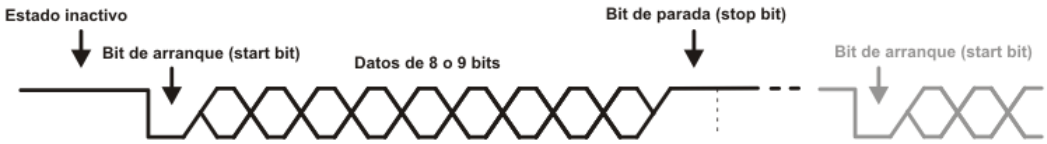


Figura 2.10 Método de Transmisión Serial [11]

Cada dato se transmite de la siguiente forma:

- ✓ En estado inactivo la línea de datos permanece en estado alto (1).

- ✓ Cada transmisión de datos comienza con un bit de arranque (START), el cual, siempre es cero (0).
- ✓ Cada dato tiene un ancho de 8 o 9 bits (primero se transmite el bit menos significativo).
- ✓ Cada transmisión de datos termina con un bit de parada (STOP), el cual, siempre es uno (1).

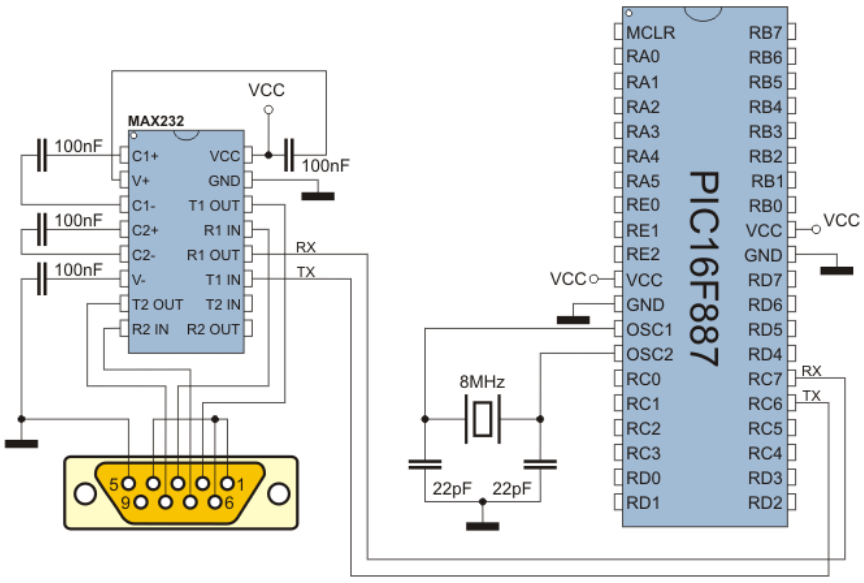


Figura 2.11 Esquemático de conexión RS232 [11]

2.5. TECNOLOGÍAS INALÁMBRICAS

2.5.1. PROTOCOLO ZIGBEE

Es el nombre de la especificación de un conjunto de protocolos de alto nivel de comunicación inalámbrica para su utilización con radiodifusión digital de bajo consumo, basada en el estándar IEEE 802.15.4 de redes inalámbricas de área personal. [1]

Su objetivo son las aplicaciones que requieren comunicaciones seguras con baja tasa de envío de datos y maximización de la vida útil de sus baterías.

Mayormente se encuentran aplicaciones desarrolladas en el área de domótica, la razón de ello:

- ✓ Su fácil integración.
- ✓ El bajo consumo de energía.
- ✓ Su topología de red en malla.

Una de sus características importantes es que el protocolo ZigBee usa la banda ISM (Industrial, Scientific and Medical), en concreto, 898MHz en Europa, 915 MHz en Estados Unidos y 2.4 GHz en todo el mundo, y para este proyecto se optó por la frecuencia de 2.4 GHz por ser libre en todo el mundo.

2.5.2. COORDINADOR ZIGBEE (ZC)

Debe existir uno por red que se encargue de controlar la red y los caminos que deben seguir los dispositivos para conectarse entre ellos. Es el dispositivo más completo.

2.5.3. ROUTER ZIGBEE (ZR)

Interconecta dispositivos que se encuentran separados en la topología de la red, además de ofrecer una capa de aplicación para la ejecución de código de usuario.

2.5.4. DISPOSITIVO FINAL (ZED)

Este dispositivo posee la funcionalidad necesaria para comunicarse con un nodo padre o router, pero no transmite información a otros dispositivos. De esta manera el nodo puede estar dormido la mayor parte del tiempo ayudando a la vida media de la batería. Es relativamente barato puesto a que no tiene grandes requerimientos de memoria.

Según la funcionalidad de los dispositivos se clasifican en:

DISPOSITIVO DE FUNCIONALIDAD COMPLETA (FFD)

Son conocidos también como nodos activos, son capaces de recibir mensajes en formato 802.15.4, pueden funcionar como Router o

Coordinador, son usados en dispositivos de red donde actúe de interface con el usuario.

DISPOSITIVO DE FUNCIONALIDAD REDUCIDA (RFD)

Llamado también nodo pasivo, puesto que tiene capacidad y funcionalidad limitada. En sí son los sensores o actuadores de la red.

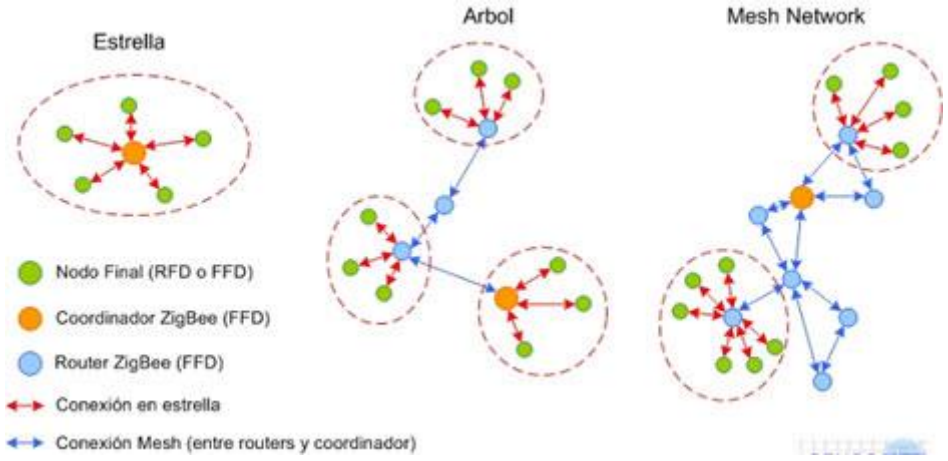


Figura 2.12 Modelo de Red ZigBee [10]

MÓDULOS DE COMUNICACIÓN INALÁMBRICA XBEE

Los módulos Xbee proveen 2 formas amigables de comunicación, como lo son la transmisión serial transparente (modo AT) y el modo API. Los módulos Xbee pueden ser configurados desde la PC y estos pueden comunicarse en arquitecturas punto a punto, punto a multipunto o en una red mesh. [1]

Figura 2.13 Módulo Xbee [1]

Aplicaciones:

Los módulos Xbee, pueden ser ajustados para usarse en redes de configuración punto a punto, punto- multipunto o peer-to-peer.

Con los módulos Xbee PRO de la Serie 2, es posible crear redes más complejas, como las llamadas MESH que es la que utilizamos en nuestro proyecto. Estas redes permiten acceder a un punto remoto, utilizando módulos intermedios para llegar como routers.

Características de los Xbee:

- ✓ Alimentación de 3.3 VDC.
- ✓ Velocidad de transmisión máxima de 250 Kbps.

- ✓ Rango de transmisión de 120m en línea de vista y 40 con obstáculos.
- ✓ 6 entradas analógicas para conversión ADC a una resolución de 10 bits.
- ✓ 8 entradas digitales.

2.6. HERRRAMIENTAS DE DESARROLLO ASISTIDA POR COMPUTADOR.

2.6.1. QUARTUS II

Quartus II es una herramienta de software producida por Altera para el análisis y la síntesis de diseños realizados en HDL.

Entre las funciones que permite realizar Quartus II al desarrollador tenemos el poder compilar sus diseños, realizar análisis temporales, analizar circuitos lógicos y configurar el dispositivo destino con el programador.

Quartus II nos permite realizar varias tareas al momento de la simulación de un circuito lógico, desde la descripción de los componentes del circuito en lenguaje VHDL, hasta la simulación de diagramas de tiempo del comportamiento del circuito. Además nos ofrece muchas ventajas al momento de desarrollar un proyecto usando

el microprocesador NIOS II ya que con la herramienta integrada Qsys podemos generar el Hardware que requerimos para la implementación.

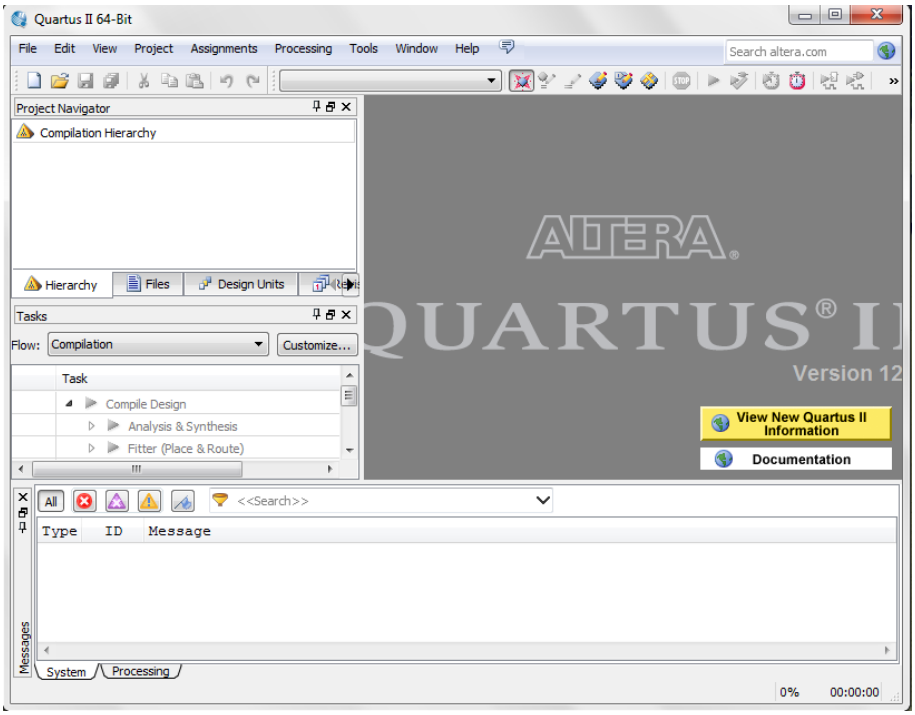


Figura 2.14 Ventana de trabajo de QUARTUS II

2.6.2. ECLIPSE

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma, típicamente ha sido usada para desarrollar entornos de desarrollo integrados (del inglés IDE), como el IDE de Java llamado Java Development Toolkit (JDT) y el

compilador (ECJ) que se entrega como parte de Eclipse (y que son usados también para desarrollar el mismo Eclipse). Sin embargo, también se puede usar para otros tipos de aplicaciones cliente, como BitTorrent o Azureus. [6]

Características:

Eclipse se encuentra estructurado de la siguiente manera:

- ✓ Plataforma principal para el inicio de Eclipse y ejecución de Plug-in's.
- ✓ OSGi, permite diseñar plataformas compatibles que puedan proporcionar múltiples servicios.
- ✓ El SWT, que es un Widget Toolkit portable.
- ✓ JFace, para manejo de archivos, manejos de texto, editores de texto, etc.
- ✓ Workbench de Eclipse, para vistas, editores, perspectivas, asistentes, etc.

Programación:

Este software usa lenguajes de programación como C/C++ y Phython, además de permitir trabajar con lenguajes para procesamiento de texto como LaTeX, aplicaciones de red como Telnet y sistema de gestión de base de datos, además de proveer al programador con frameworks para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, etc.

2.6.3. PROTEUS

Es una compilación de programas de diseño y simulación electrónica, desarrollado por Labcenter Electronics que consta de dos programas principales: Ares e ISIS, y el módulo VSM.

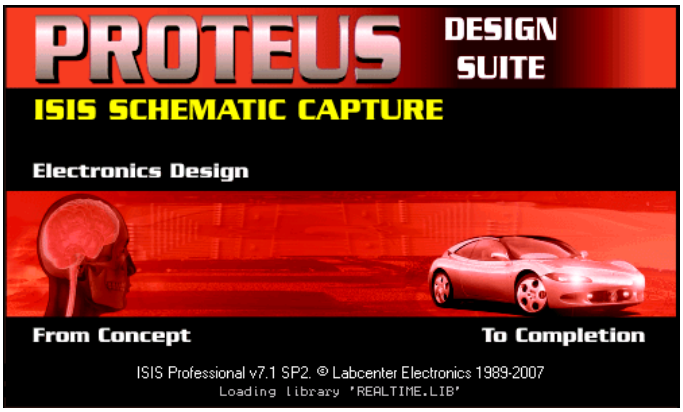


Figura 2.15 DESIGN SUITE PROTEUS [1]

ISIS (Intelligent Schematic Input System): módulo de captura de esquemas.

VSM (Virtual System Modelling): módulo de simulación.

ARES (Advanced Routing Modelling): módulo para realización de circuitos impresos (PCB)

El programa ISIS es un programa que nos permite dibujar sobre un área de trabajo un circuito que posteriormente podremos simular.

2.6.4. MIKROC

MikroC PRO for PIC organiza las aplicaciones en los proyectos que consisten en un solo fichero con extensión .mcppi en uno o más ficheros fuentes con extensión .c. Los ficheros fuentes son denominados cabeceras en lenguaje de programación C. El compilador mikroC for PIC permite manejar varios proyectos a la vez. Los ficheros fuentes se pueden compilar sólo si forman parte del proyecto. [12]

Un proyecto de fichero contiene lo siguiente:

- ✓ Nombre del proyecto y la descripción opcional.
- ✓ Tipo de microcontrolador.
- ✓ Frecuencia de reloj del microcontrolador.

- ✓ Lista de ficheros fuentes del proyecto.
- ✓ Ficheros binarios (*.mcl)

Figura 2.16 Software de Programación MIKROC [12]

2.6.5. XCTU

El programa X-CTU es una herramienta gratuita desarrollado por Digi para la configuración y manejo de sus dispositivos XBee. Con este software además de configurar los dispositivos con los parámetros adecuados para la red, es posible realizar pruebas de cobertura antes de hacer el despliegue de los dispositivos. [12]

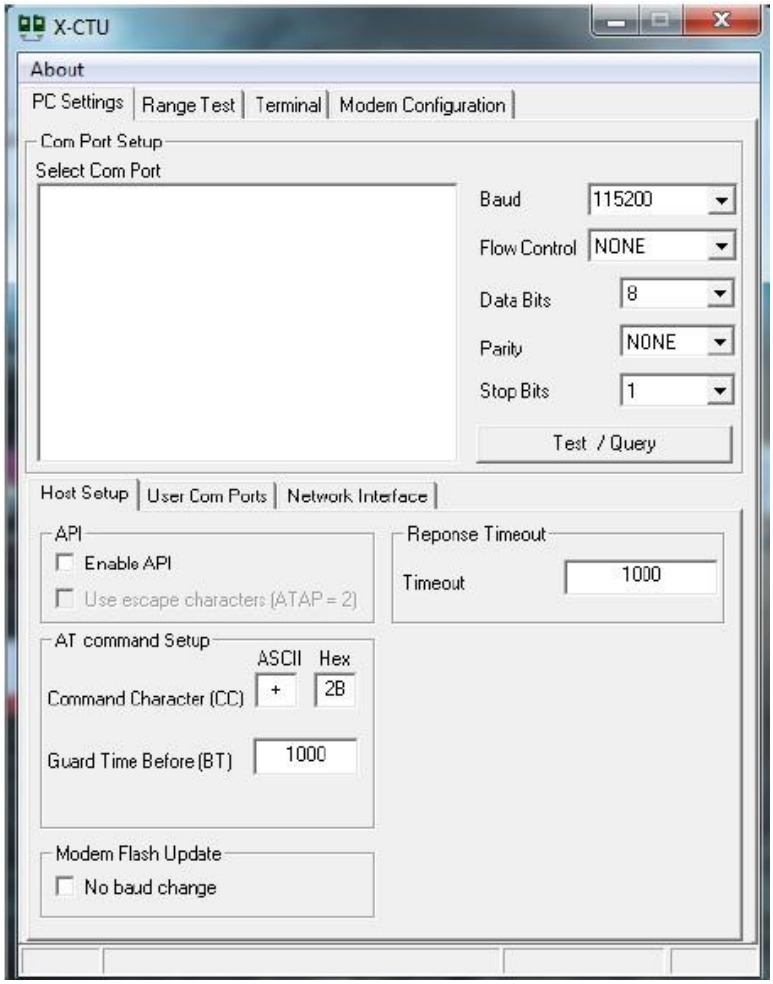


Figura 2.17 Software de Configuración de módulos Xbee X-CTU [13]

2.7. INVERNADEROS

Un invernadero (o invernáculo) es un lugar cerrado, estático y accesible a pie, que se destina a la producción de cultivos, dotado habitualmente de una cubierta exterior translúcida de vidrio o plástico, que permite el control de la temperatura, la humedad y otros factores ambientales para favorecer el desarrollo de las plantas. [7]

2.7.1. IMPORTANCIA

La producción de cultivos en invernadero es una técnica moderna usada en la producción agrícola, su ventaja sobre el método tradicional abierto es que se establece una barrera entre el medio ambiente externo y el cultivo. Dicha barrera crea un microclima que permite protegerlo del viento, granizo, sol, heladas, plagas, enfermedades, hierbas y animales, esto es lo que lo hace importante al momento de tomar una decisión al empezar a cultivar algún producto.

2.7.2. ESTRUCTURA DEL INVERNADERO

La estructura es el armazón del invernadero, el cual está constituido por vigas y soportes, que sostienen la cubierta que suele ser de plástico o de vidrio traslucido, la cual soporta el viento, la lluvia y los dispositivos que se instalan sobre las plantas, instalaciones de riego y atomización de agua, por otro lado debe haber un mínimo de sombra y libertad de movimiento interno.

La estructura del invernadero deber cumplir con las siguientes condiciones:

- ✓ Debe ser ligera, resistente y ocupar poca superficie.
- ✓ Estar construida con material económico y de fácil conservación.

- ✓ Debe ser susceptible de poder ser ampliada.

En nuestro caso construimos un invernadero tipo capilla, fabricado en vidrio, con un ventilador en la parte posterior y una fluorescente en la parte superior del techo. Por el hecho de estar elaborado en modelo capilla nos brinda la facilidad de tener una evacuación fácil del agua lluvia, para esto la inclinación del techo es de 25°.

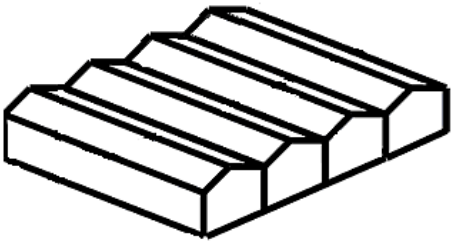


Figura 2.18 Invernadero tipo Capilla [8]

2.7.3. SISTEMA DE RIEGO

El sistema de riego implementado en nuestro invernadero es el goteo. Este sistema es uno de los más utilizados y eficientes ya que a diferencia del riego tradicional y de la aspersión, aquí el agua se conduce desde la fuente de abastecimiento a través de tuberías pinchadas y en su destino se libera gota a gota justo en el lugar donde se ubica la planta, permitiendo que el agua se infiltre en el suelo produciendo una zona húmeda necesaria en la zona radicular de cada planta.

2.8. SENSORES

Los sensores son dispositivos sensibles que utilizan un fenómeno físico o químico dependiente de la naturaleza y el valor de la magnitud físico química a medir, lo cual permite la transducción del estímulo a una señal utilizada directa o indirectamente como medida. En este proyecto se utilizan sensores de humedad, temperatura y luminosidad, de los cuales a continuación se dan algunas características de su funcionamiento.

2.8.1. SENSOR DE TEMPERATURA

La temperatura es una medida del calor o energía térmica que establece un equilibrio entre dos cuerpos. El calor fluye del cuerpo con mayor temperatura hacia el de menor temperatura. Pese a que la unidad patrón de la temperatura es el Kelvin (ºK) también se suele expresar en grados Celsius (ºC). En la práctica existen numerosos tipos de sensores de temperatura o termómetros que, según la aplicación específica, pueden ser los adecuados. [8]

Sensor DS18B20

El DS18B20 es un sensor de temperatura con una precisión de 9 a 12 bits, desde -55°C hasta 125°C (+/-0.5C). Cada sensor tiene un número serial único de 64-Bit grabado.

Características:

- ✓ Interfaz de 1-Wire para comunicación.
- ✓ Cada dispositivo tiene un código serial único de 64-bit guardado en una memoria ROM interna.
- ✓ No requiere componentes externos.
- ✓ Puede ser alimentado a través de la línea de datos. El rango de voltajes de alimentación es de 3V a 5.5V
- ✓ Realiza mediciones desde –55°C hasta +125°C (–67°F hasta +257°F)
- ✓ ±0.5°C de exactitud desde–10°C hasta +85°C
- ✓ Resolución seleccionable de 9 a 12 bits.
- ✓ Convierte la temperatura a una palabra digital de 12 bits en 750ms

- ✓ Aplicaciones de control térmico, sistemas industriales, productos finales, termómetros y cualquier otro sistema que sea sensible térmicamente. [14]

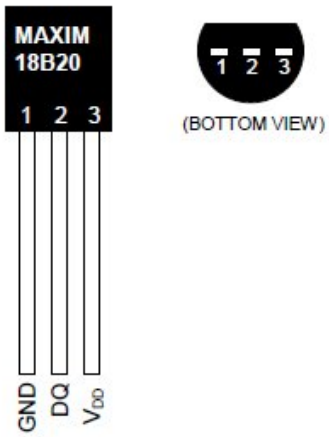


Figura 2.19 Pines de Sensor de Temperatura DS18B20 [15]

2.8.2. SENSOR DE HUMEDAD

Las especificaciones del sensor SKU: SEN0114 son:

- ✓ Alimentación de 3.3 o 5 Voltios.
- ✓ Voltaje de salida: 0 a 4.2 Voltios.
- ✓ Corriente: 35 mA.

Pines de funcionamiento:

- ✓ Señal de salida: HUMEDAD.
- ✓ Tierra de alimentación.
- ✓ Suministro de energía (+5.0 V).

Rango de Humedad:

SALIDA EN VOLTIOS	ESTADO DEL SUELO
0	Suelo completamente húmedo
1 – 2	Suelo Húmedo
4	Suelo Seco

Tabla 2.1 Relación Voltaje vs. Humedad del Suelo

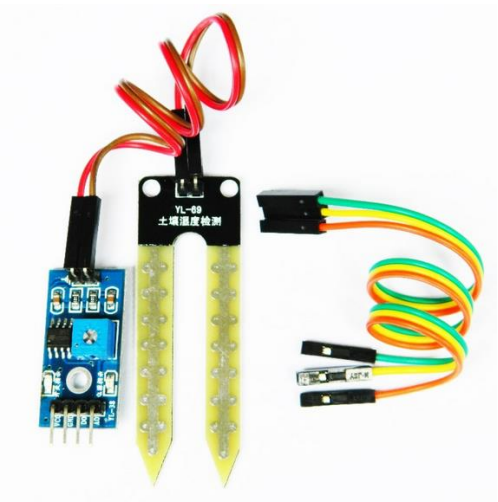


Figura 2.20 Sensor de Humedad SKU: SEN0114

2.8.3. SENSOR DE LUMINOSIDAD

La fotorresistencia es una resistencia cuyo valor dependen de la energía luminosa incidente en ella, específicamente son resistencias cuyo valor de resistividad disminuye a medida que aumenta la energía luminosa incidente sobre ella y viceversa. Una fotorresistencia se compone de un material semiconductor cuya resistencia varía en función de la iluminación. La fotorresistencia reduce su valor resistivo en presencia de rayos luminosos. Es por ello por lo que también se le llama resistencias dependientes de luz (LDR - Light Dependent Resistors).

Los valores que puede tomar una LDR en total oscuridad y a plena luz puede variar un poco de un modelo a otro, en general oscilan entre unos

50 a 1000 ohmios (1K) cuando están iluminadas (por ejemplo, con luz solar) y valores comprendidos entre 50K (50,000 Ohms) y varios mega ohmios (millones de Ohms) cuando está a oscuras.

Rango de Luminosidad:

SALIDA EN VOLTIOS	LUMINOSIDAD
0 - 1 [V]	Día
1.1 – 3 [V]	Tarde
3.1 – 5 [V]	Noche

Tabla 2.2 Relación Voltaje vs. Luminosidad.

CAPÍTULO 3

3. DISEÑO E IMPLEMENTACIÓN

En este capítulo se muestra cómo se diseñó e implementó el proyecto, así como también los parámetros utilizados en la elaboración del mismo.

3.1. ANÁLSIS DE REQUERIMIENTOS

Para llevar a cabo el diseño e implementación de la red de sensores Inalámbricos debemos tomar en cuenta las especificaciones que mencionamos al comienzo, las cuales nos servirán de guía a lo largo de todo el proceso.

Especificaciones:

- ✓ Como núcleo central del proyecto tenemos al microprocesador NIOS II.
- ✓ Generar un menú amigable con el usuario para que pueda ingresar las especificaciones del cultivo a monitorear.
- ✓ La información es obtenida de diferentes sensores.
- ✓ Implementar un protocolo de comunicación para la transmisión y recepción de datos entre el nodo central y los nodos sensores.
- ✓ Visualizar los datos recibidos.
- ✓ Almacenar la información obtenida en una memoria SD Card.
- ✓ Los datos almacenados deben estar en formato .xls, formato por defecto que genera la aplicación Excel, en donde encontraremos una lista de las mediciones de humedad vs. temperatura, mediante la cual podemos generar un gráfico del mismo para que el usuario pueda interpretarlo de manera didáctica.

Para la obtención de la información proporcionada por los sensores utilizamos un microcontrolador de la familia microchip PIC 16F887, el mismo que posee los módulos requeridos para el procesamiento de la información que será enviada al microprocesador NIOS II. Dicho envío de datos se lleva a cabo por medio de módulos de comunicación inalámbrica

Xbee Serie 2, los cuales utilizan el protocolo Zigbee para el intercambio de información.

Debido a que el microprocesador NIOS II se encuentra embebido en una FPGA integrada en la tarjeta de desarrollo DE2-115 junto con los periféricos de entrada y salida de datos es necesario implementar un protocolo de comunicación entre el microcontrolador y la tarjeta DE2-115. El protocolo utilizado fue el RS232 puesto que nos proporciona estándares previamente establecidos como: velocidad de transferencia de datos, control que utiliza dicha transferencia, niveles de voltaje, conectores, entre otros.

Una vez recibidos los datos por la tarjeta DE2-115 son visualizados en tiempo real en una pantalla LCD que se encuentra integrada en la misma y almacenados en una tarjeta SD creando una base de datos que es transformada en formato .xls para mayor facilidad de interpretación del usuario

3.2. DIAGRAMA FUNCIONAL

En la figura 3.1 se muestra un diagrama funcional de nuestro proyecto, el cual describe cada etapa del proceso del mismo, a continuación se da una breve explicación del funcionamiento del mismo.

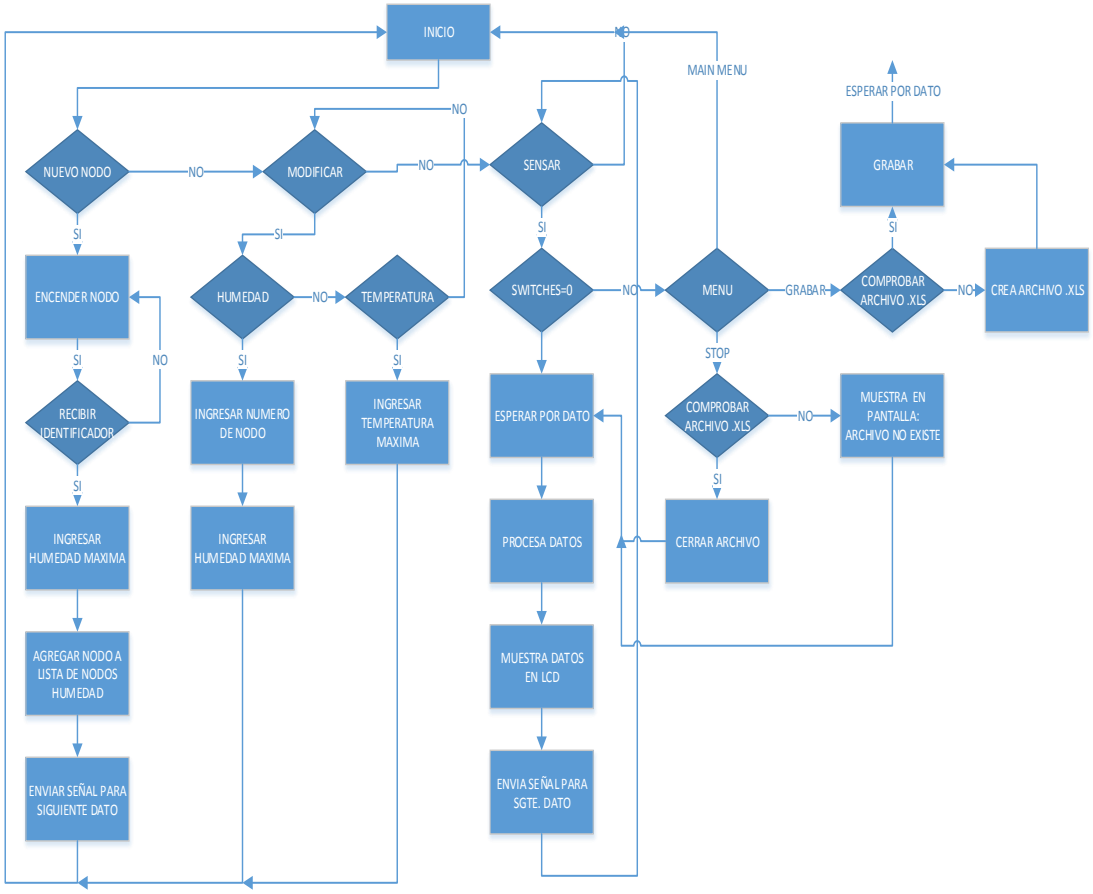


Figura 3.1 Diagrama Funcional de la solución implementada

Una vez iniciado el sistema, el usuario debe seleccionar una de las tres opciones en el menú. En el caso de seleccionar la opción NUEVO NODO deberá encender el nodo sensor de humedad para que se sincronice con la tarjeta e ingresar las especificaciones apropiadas para el cultivo a monitorear, de esta manera se agrega el nodo sensor a la lista de Sensores de Humedad, al finalizar el proceso envía un identificador al nodo para que empiece a sensar; este proceso no se hace con los nodos de Temperatura

y Humedad puesto que sólo existe uno de cada tipo por sistema y se emparejan automáticamente al encenderse.

En la segunda opción se puede modificar las condiciones de monitoreo del cultivo tales como humedad y temperatura. En el caso de ser humedad se debe ingresar el número de nodo a modificar, el cual es el identificador que se le da a cada nodo en el orden en el que se encienden. Por otro lado si es temperatura se pide al usuario ingresar la temperatura máxima para el cultivo en general.

En la opción de SENSAR los sensores empiezan a realizar sus mediciones en tiempo real, se muestra en la pantalla LCD los datos obtenidos.

Durante cualquier proceso el usuario puede activar un switch para acceder a un menú en el cual se brinda al usuario la posibilidad de grabar en una memoria SD Card un archivo .xls de nombre Monitoreo y detener la grabación cerrando el mismo archivo, o regresar al menú principal.

3.3. DIAGRAMA DE BLOQUES

Para demostrar de manera más sencilla el funcionamiento de la red inalámbrica de sensores se construyó un diagrama de bloque, el cual lo mostramos en la figura 3.2 y se puede observar lo siguiente:

- ✓ Existen 3 nodos sensores transmisores de los cuales dos son analógicos: humedad y luminosidad, y uno digital: temperatura, y un nodo central (Tarjeta de Desarrollo DE2-115), cada uno con su módulo transmisor Xbee Serie 2.

- ✓ Existe una etapa de digitalización por medio del módulo ADC presente en el PIC 16F887.

- ✓ Recepción de los valores transmitidos por el PIC en la tarjeta DE2-115.

- ✓ Visualización de los datos recibidos por los sensores en la pantalla LCD de la tarjeta DE2-115.

- ✓ Almacenamiento de los datos procesados en una memoria SD Card.

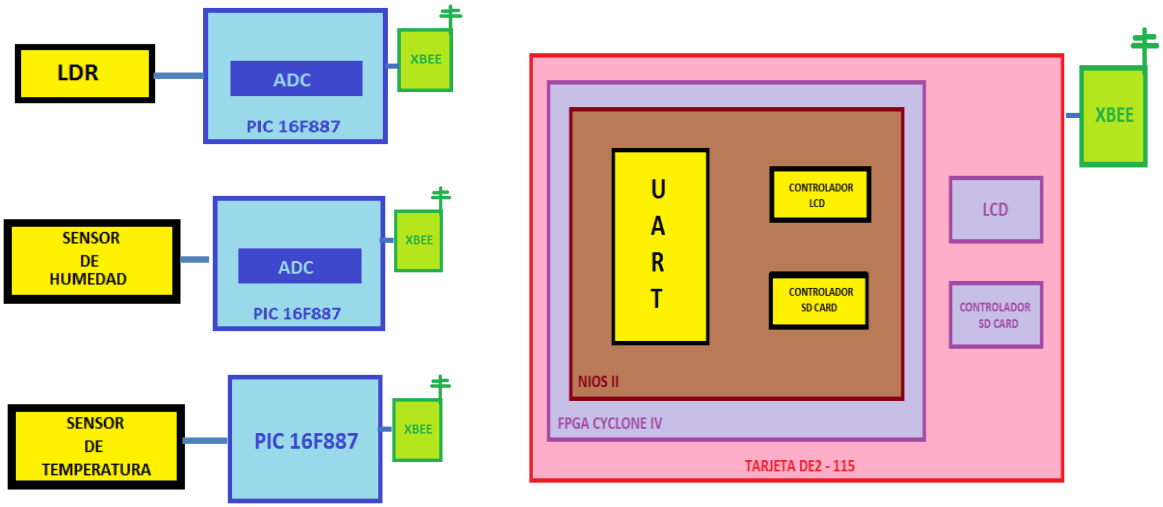


Figura 3.2 Diagrama de Bloques de la Red Inalámbrica de Sensores

En el diagrama de bloques se puede ver claramente que existen dos etapas bien definidas, una conformada por los nodos sensores transmisores mediante el PIC 16F887 y otra etapa conformada por la tarjeta DE2-115, la cual es parte del módulo receptor y el nodo central quien es el encargado de tomar las decisiones.

3.4. ESQUEMÁTICO DE LOS NODOS SENSORES

Para el diseño del hardware de los nodos sensores que son los módulos transmisores se han tomado las siguientes consideraciones:

- ✓ Obtener los valores analógicos de humedad y luminosidad.
- ✓ Obtener los valores digitales de temperatura.
- ✓ Digitalizar valores analógicos según sea el caso.
- ✓ Transmisión inalámbrica de los datos sensados mediante los módulos Xbee.

A continuación se muestra el esquemático y el PCB que se realizó para llevar a cabo la implementación de cada uno de los módulos transmisores. Teniendo en cuenta que cada uno está conformado por 4 etapas:

- ✓ Alimentación

- ✓ Sensores
- ✓ Red de estabilidad
- ✓ Digitalización – Transmisión realizada con el PIC 16F887.

En la figura 3.3 se muestra el esquemático de los 3 módulos sensores, debido a que en la programación del PIC16F887 utilizamos la misma asignación de pines exceptuando por los sensores analógicos que utilizan las entradas analógicas, realizamos un solo esquemático básico.

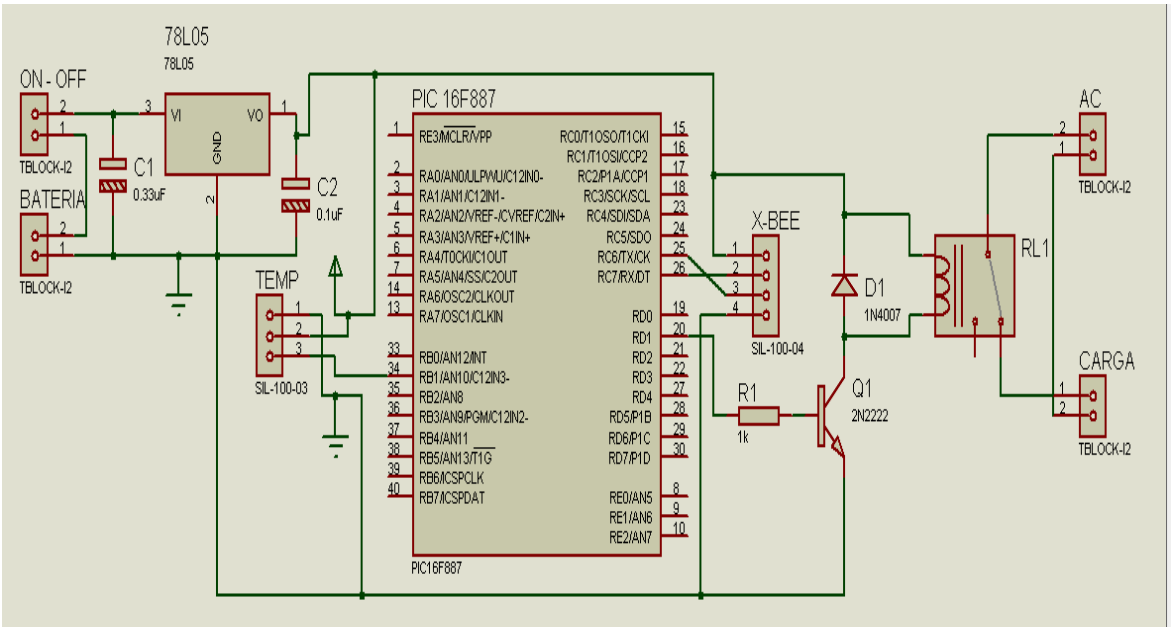


Figura 3.3 Esquemático del Módulo Transmisor

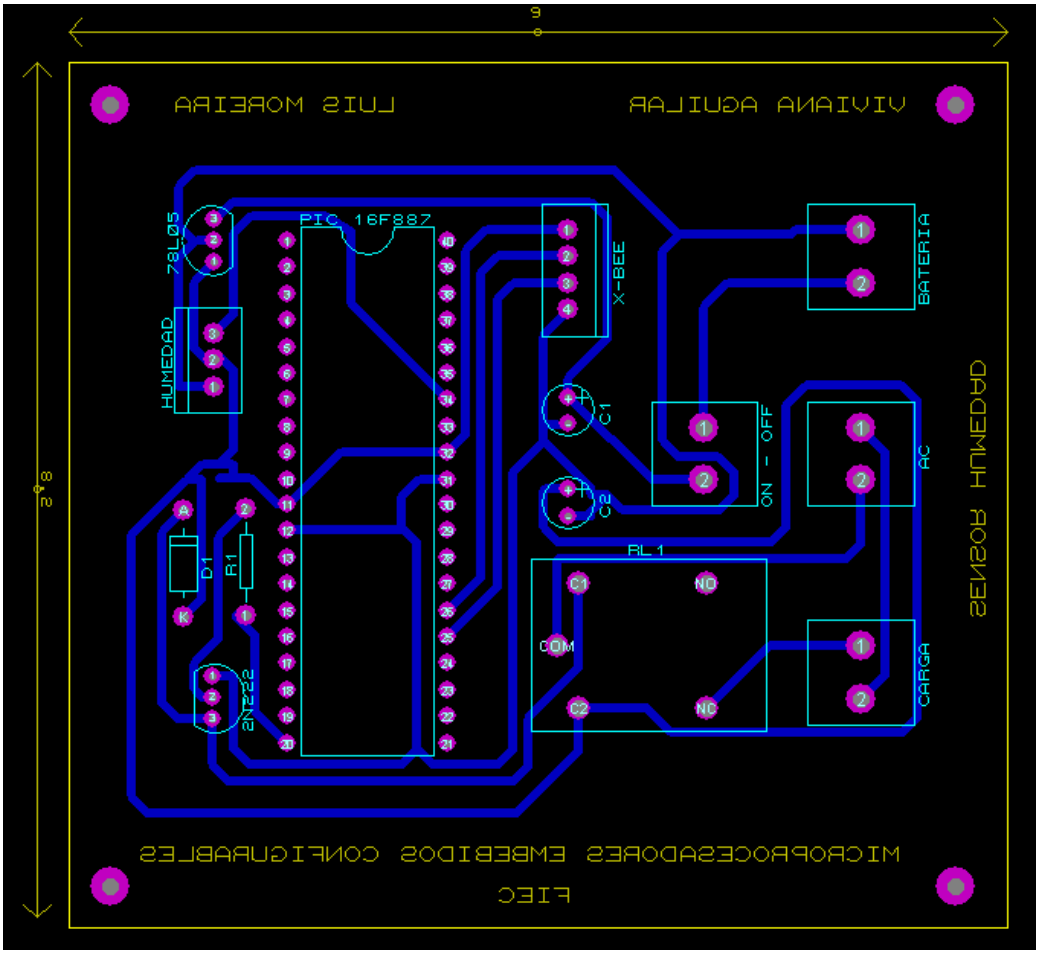


Figura 3.4 PCB del Módulo Transmisor

3.5. DISEÑO DEL HARDWARE – MÓDULO RECEPTOR EN QSYS

Para la implementación del proyecto se trabajó sobre la máquina media de la tarjeta DE2-115 de Altera y por medio de la herramienta Qsys del Quartus II se agregaron el módulo SD Card y el UART y se eliminaron los módulos no utilizados.

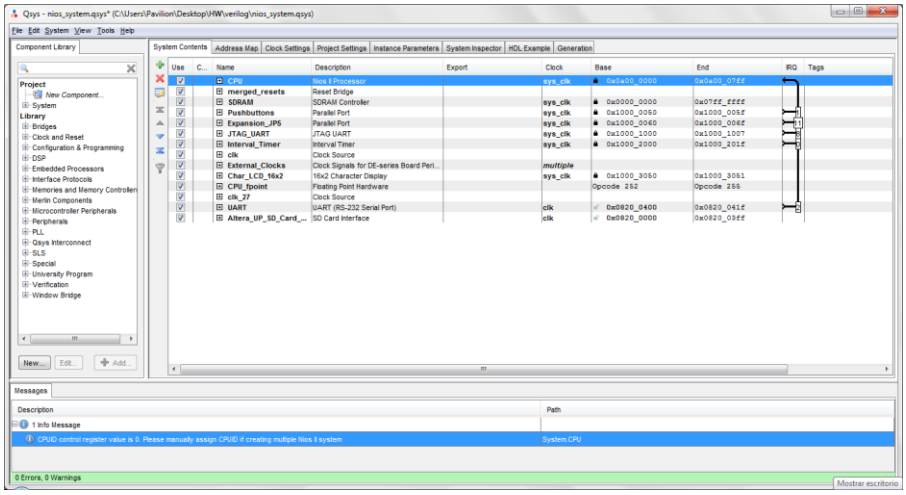


Figura 3.5 Diseño del Módulo Receptor en Qsys.

3.5.1. MICROPROCESADOR NIOS II

Todo sistema embebido debe ser controlado mediante un microprocesador el cual es el encargado de interactuar con las interfaces conectadas a él, procesar la información que recibe y generar las diferentes señales de control para cada interfaz.

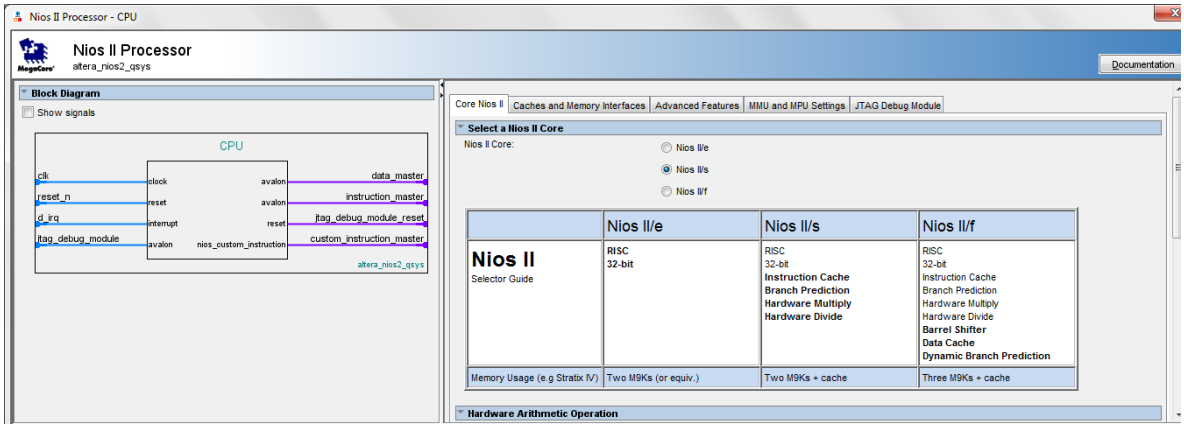


Figura 3.6 Configuración del Microprocesador NIOS II

3.5.2. UART RS232

La interfaz serial es la que permite el envío y recepción de datos mediante el módulo XBEE. Entre sus características importantes tenemos:

- ✓ Velocidad de transmisión a 9600 Bps.
- ✓ 8 Bits para los datos.
- ✓ 1 Bit de parada.

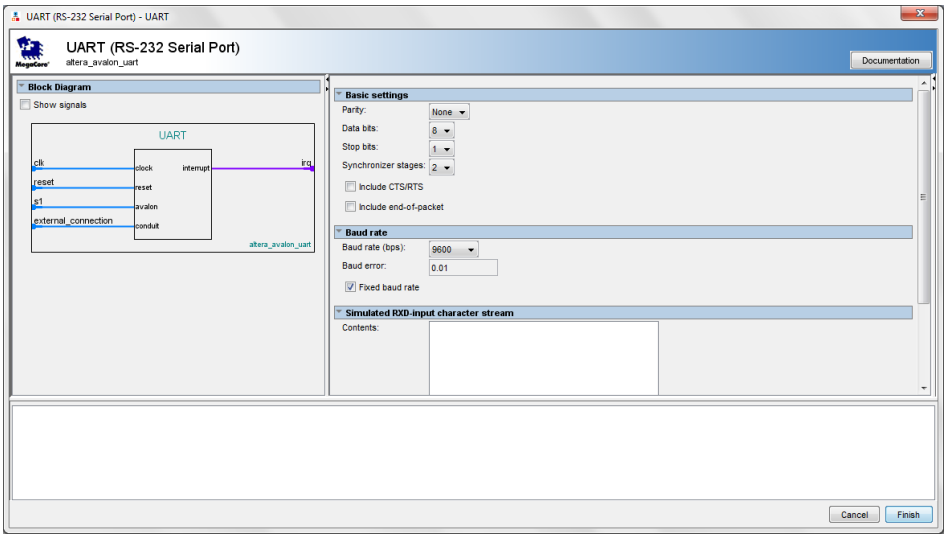


Figura 3.7 Configuración del Módulo UART.

3.5.3. LCD 16X2

La visualización de los datos que recibe la tarjeta DE2-115 se realiza mediante la pantalla LCD, el control de este procedimiento se lo realiza por medio del procesador NIOS II en conjunto con el módulo SDRAM.

Para controlarlo hacemos uso del software NIOS II usando las librerías que se generan una vez terminado el diseño en Qsys.

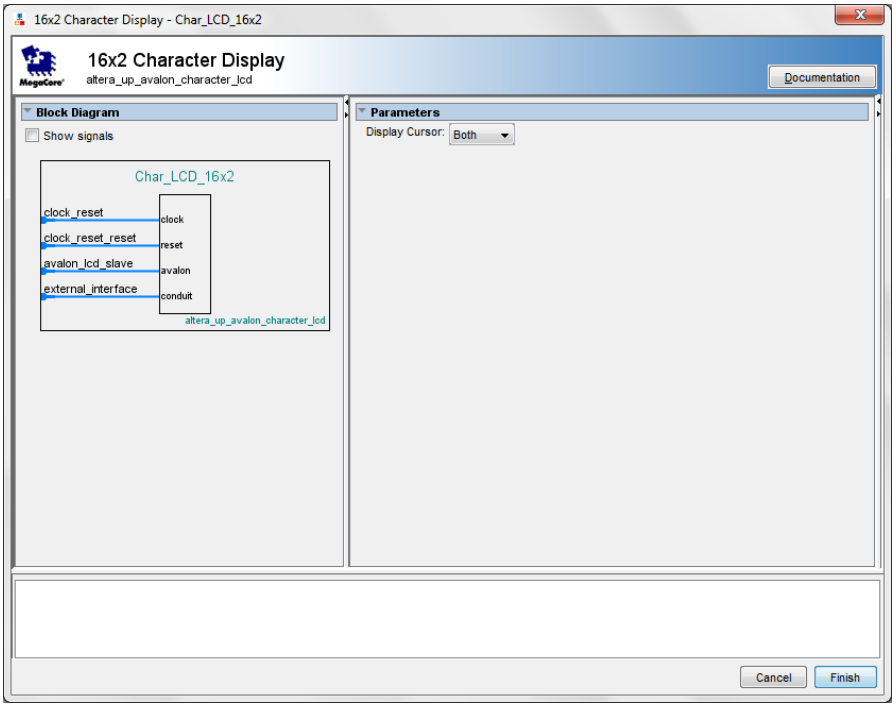


Figura 3.8 Configuración del Módulo LCD 16x2

3.5.4. BOTONERAS

Mediante las botoneras podemos acceder a las opciones que presenta el menú, haciendo uso de las 3 primeras botoneras podemos seleccionar cualquiera de las 3 opciones, la cuarta botonera nos servirá para resetear el programa.

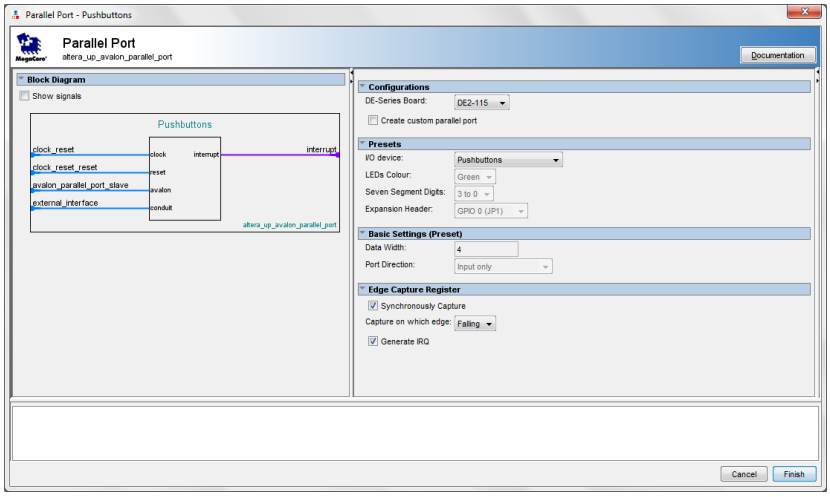


Figura 3.9 Configuración para Botoneras

3.5.5. SD CARD

Para poder realizar el almacenamiento de datos se usa una tarjeta SD, la misma que se debe insertar en el conector SD Card presente en la tarjeta DE2-115. A nivel de software se establecen los comandos que debe ejecutar el procesador NIOS II para realizar la escritura de una base de datos del monitoreo del invernadero.

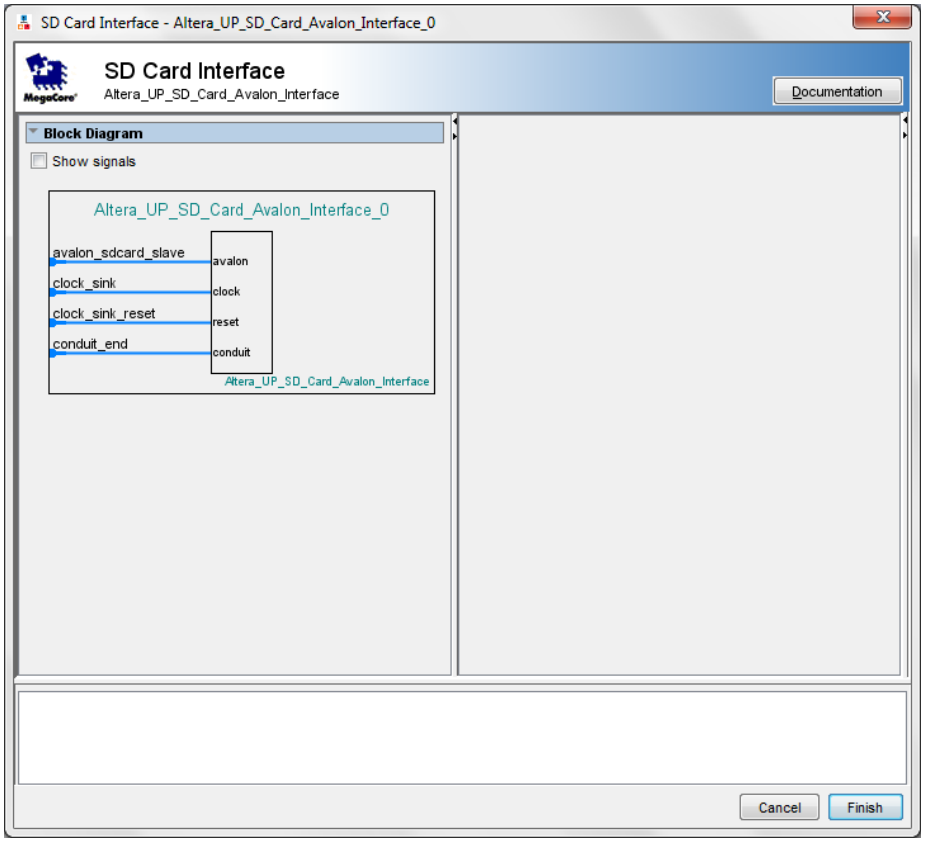


Figura 3.10 Configuración para Interfaz SD Card.

3.5.6. SWITCHES DESLIZANTES

Los switches implementados en nuestro sistema son utilizados para indicarle al procesador NIOS II que debe enviar señales de control a la interfaz SD Card y comenzar con el almacenamiento de los datos recibidos.

3.5.7. PUERTO DE EXPANSIÓN JP5

El puerto de expansión JP5 está conformado por 36 pines de propósito general, 2 pines GND, 1 pin LV-TTL y 1 pin TTL. Los 36 pines de propósito general pueden ser configurados como entradas o salidas de las cuales hemos utilizado los pines: 8 (AD 15) y 9 (AE 15) como TX y RX para conectar el módulo Xbee y los pines 10 y 11 para alimentar el módulo.

3.6. PROGRAMACIÓN EN LENGUAJE C USANDO MIKRO C Y ECLIPSE.

A continuación se muestran las funciones utilizadas en la programación de ambos procesadores, estas funciones son necesarias para la implementación del código de la solución de nuestro proyecto.

Programación PIC

UART1_Init

Esta función permite inicializar el módulo UART para la transmisión de datos a una velocidad seleccionada por el programador en baudios por segundos.

UART1_Write

Esta función permite enviar los datos digitalizados por el pin TX del microcontrolador.

ADC_READ

Función que permite leer el valor analógico presente en cualquier pin configurado como entrada analógica del microcontrolador y convierte este valor analógico en un valor digital.

Programación de Procesador NIOS II

El programa Universitario de Altera nos proporciona una serie de librerías para cada módulo que se conecta con el procesador NIOS II. Adicional a esto fue necesario implementar funciones para cumplir con todas las especificaciones de nuestro proyecto, las cuales se detallan a continuación:

3.6.1. PANTALLA LCD

Las librerías utilizadas para poder usar la pantalla LCD fueron:

altera_up_avalon_character_lcd.h

altera_up_avalon_character_lcd_regs.h

Dichas librerías contienen las funciones básicas y necesarias para poder manejar la pantalla LCD mediante el procesador NIOS II.

Funciones implementadas para la solución de nuestro proyecto:

void LCD_cursor (int x, int y)

Función que permite posicionar el cursor de la LCD en cierta dirección especificada mediante las variables enteras X y Y.

Void LCD_text (char* text_ptr)

Esta función es utilizada para poder escribir en la pantalla LCD, es utilizada después de posicionar el cursor en la LCD.

Void LCD_cursor_off (void)

Función que sirve para apagar el cursor de la pantalla LCD y poder visualizar mejor los datos.

void LCD_clear_screen (alt_up_character_lcd_dev *lcd, int len)

Esta función fue implementada para borrar el contenido que se está visualizando en la pantalla LCD, dependiendo del valor que posea la variable len.

Char* convertintChar (int num)

Esta función permite convertir los datos de tipo char a tipo entero.

void subir_bajar (alt_up_character_lcd_dev *lcd, int * referencia)

Función que utilizamos para poder variar un valor de referencia especificado, haciendo uso de las dos primeras botoneras, tanto para subir el valor o para disminuir el mismo.

3.6.2. UART RS232

Para el módulo UART se usaron las siguientes librerías:

altera_up_avalon_rs2323.h

altera_up_avalon_rs232_regs.h

Dichas librerías contienen la descripción de funciones necesarias para el manejo del módulo UART mediante el procesador NIOS II.

3.6.3. SDCARD

Para poder hacer uso de la SD CARD utilizamos la librería:

altera_up_sd_card_avalon_interface.h

Esta librería contiene las funciones necesarias para el manejo del módulo SD CARD mediante el procesador NIOS II.

A continuación se detallan algunas funciones utilizadas para la creación y almacenamiento de un archivo en la SD CARD:

short int alt_up_sd_card_fopen(char * name, bool create)

Esta función permite crear el archivo enviando el nombre y la extensión del mismo como parámetros.

bool alt_up_sd_card_write(short int archivo, char byte)

Función que permite almacenar caracter por caracter en el archivo previamente creado en la SD CARD.

3.7. IMPLEMENTACIÓN FÍSICA

Aquí se muestra la implementación física de los módulos transmisores y del módulo receptor así como su respectiva simulación.

3.7.1. SIMULACIÓN

Las simulaciones de los nodos sensores las realizamos por medio del programa PROTEUS el cual brinda un entorno amigable al momento de realizar un circuito en particular. A continuación se muestra el proceso descrito anteriormente para sensar: humedad, temperatura y luminosidad y transmitirla hacia la tarjeta DE2-115.

Cada nodo sensor está programado para enviar un dato ya sea de humedad, temperatura o luminosidad siempre y cuando éste sea diferente al anterior. En la figura 3.10 se puede visualizar que el sensor

sólo envía datos una vez que varía la temperatura, reduciendo así la cantidad de datos a procesar significativamente.

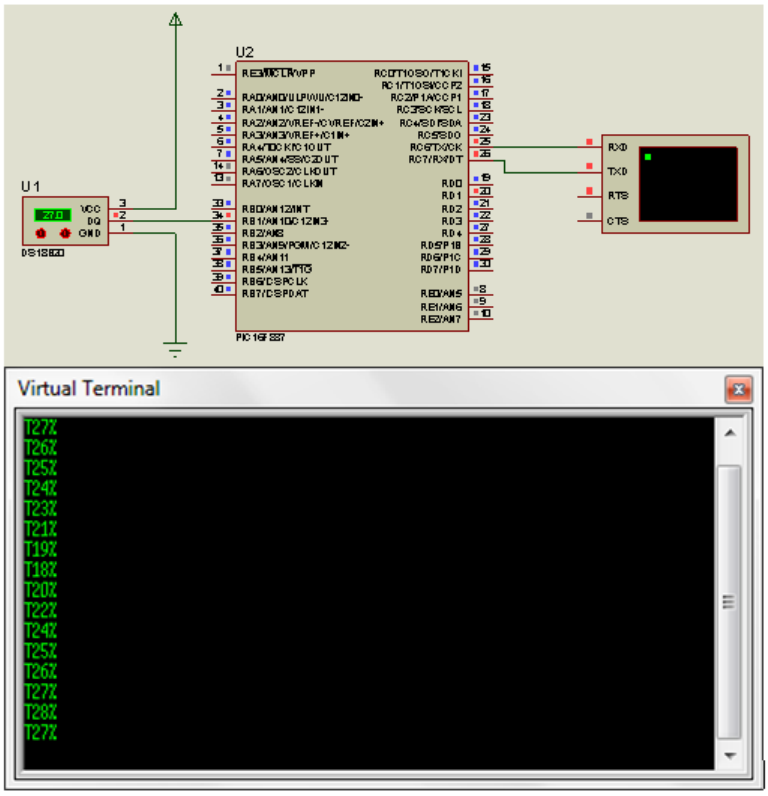


Figura 3.11 Simulación del Sensor de Temperatura.

3.7.2. COMUNICACIÓN INALAMBRICA

La transmisión de los datos sensados por los módulos transmisores hacia la tarjeta DE2-115 se realiza de forma inalámbrica por medio de módulos Xbee Serie2.

Cada nodo de la red tiene integrado un módulo Xbee el cual está configurado en modo Router, lo que nos permite enviar y recibir datos hacia el nodo coordinador como si se tratara de una conexión punto a punto.

Por otro lado el nodo central, la tarjeta DE2-115, tiene asociado un módulo Xbee en modo coordinador, lo cual permite tener comunicación en broadcast, simulando una conexión punto multipunto.

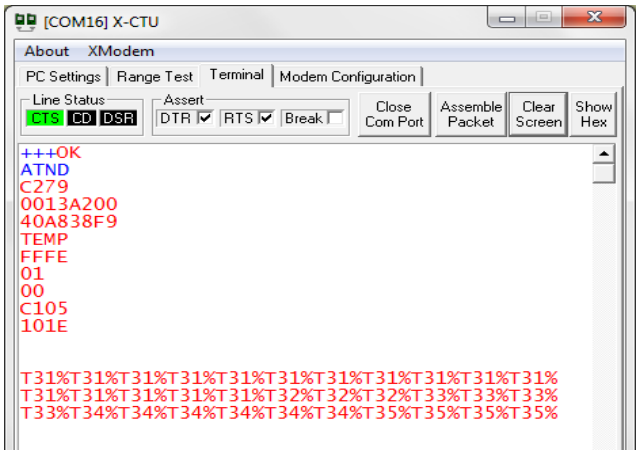


Figura 3.12 Visualización de los datos enviados desde el Sensor de Temperatura.

3.7.3. PROTOCOLO DE COMUNICACIÓN IMPLEMENTADO

Para establecer la comunicación punto a punto entre cada nodo y la tarjeta hemos creado un protocolo en el cual cada nodo tiene asociado a él dos caracteres únicos, de los cuales, uno es el nombre del sensor y

sirve para indicarle a la tarjeta que tiene un dato listo para ser enviado, y el otro establece la comunicación con la tarjeta DE2-115.

Cuando un nodo tiene un dato listo para ser enviado, primero escucha si la tarjeta está disponible para establecer comunicación, de ser así la tarjeta envía el carácter correspondiente para que proceda a enviar el dato sensado, caso contrario, es decir si la tarjeta está ocupada con otro nodo espera 4 segundos antes de anunciarse nuevamente con la tarjeta.

Por otro lado la tarjeta recibe el identificador del nodo que quiera comunicarse con ella, revisa su base de datos y envía el carácter que le corresponde para comunicarse vía punto a punto con ese nodo, mientras los demás esperan un tiempo de penalización de 4 segundos.

NODO	IDENTIFICADOR PROPIO	IDENTIFICADOR DE COMUNICACIÓN
Temperatura	T	V
Humedad	A	M
Luminosidad	L	N

Tabla 3.1 Caracteres Identificadores para cada Nodo.

En la tabla 3.1 se muestra cada uno de los identificadores que utiliza el protocolo de comunicación implementado, tanto como el identificador de comunicación y el identificador propio de cada nodo sensor.

Cada dato enviado por los nodos sensores recibe una orden en respuesta desde la tarjeta DE2-115, ésta acción tiene como objetivos principales

indicarle al nodo sensor que puede continuar sensando datos y también sirve como acuse de recibido, es decir si el nodo sensor no recibe una orden de la tarjeta DE2-115 en respuesta de su dato enviado el nodo retrasmite, ya que esto implica que la tarjeta no proceso su dato.

3.7.4. GENERACIÓN DEL ARCHIVO SOPC INFO

El archivo .SOPCINFO contienen toda la configuración sobre el sistema que se está implementando en base al procesador NIOS II, es decir contiene toda la configuración necesaria de cada módulo conectado al procesador NIOS II para luego embeber todo el sistema sobre una FPGA. Cualquier módulo que se desee agregar o eliminar del sistema antes de generar el archivo SOPCINFO debe estar previamente configurado.

CAPÍTULO 4

4. PRUEBAS Y RESULTADOS

Finalmente en este capítulo se exponen los resultados prácticos del sistema en pleno funcionamiento dentro del invernadero. Se observan los datos recopilados de cada sensor por la tarjeta DE2-115, se puede apreciar qué sucede cuando se configuran de manera errónea los parámetros del Xbee así como también la manera correcta de configurarlos. Adicionalmente el usuario tiene la opción de grabar los datos recopilados en una memoria SD Card para poder llevar un seguimiento más detallado de los diferentes cultivos.

A continuación se muestran los diferentes escenarios de pruebas realizados:

Escenario 1: Configuración errónea del módulo Xbee.

Escenario 2: Monitoreo de temperatura a lo largo de un día comparado con los datos proporcionados por una aplicación web.

Escenario 3: Medición de Frecuencia Estadística de envío de datos de cada nodo.

Escenario 4: Medición de Frecuencia Estadística de colisión de datos.

Escenario 5: Medición de Convergencia de la Red.

Escenario 6: Capacidad máxima del datalogger.

Escenario 7: Tiempo de duración de las baterías.

4.1.ESCENARIO 1: CONFIGURACIÓN ERRÓNEA DEL MÓDULO XBEE

Los módulos Xbee son dispositivos muy versátiles al momento de implementar una red tipo malla, pero existen ciertos parámetros que deben ser previamente configurados.

Para establecer la comunicación entre el Nodo sensor y la tarjeta DE2-115:

- ✓ Deben funcionar en la misma subred.
- ✓ Su configuración en cuanto a bits de parada debe coincidir.
- ✓ Deben transmitir a la misma velocidad de Baudios/Segundo.

En la figura 4.1 se muestra la configuración realizada para el nodo Router y el nodo Coordinador, con esta configuración no existe comunicación alguna entre los dispositivos de la red ya que los módulos Xbee se encuentran operando en canales y subredes distintas.

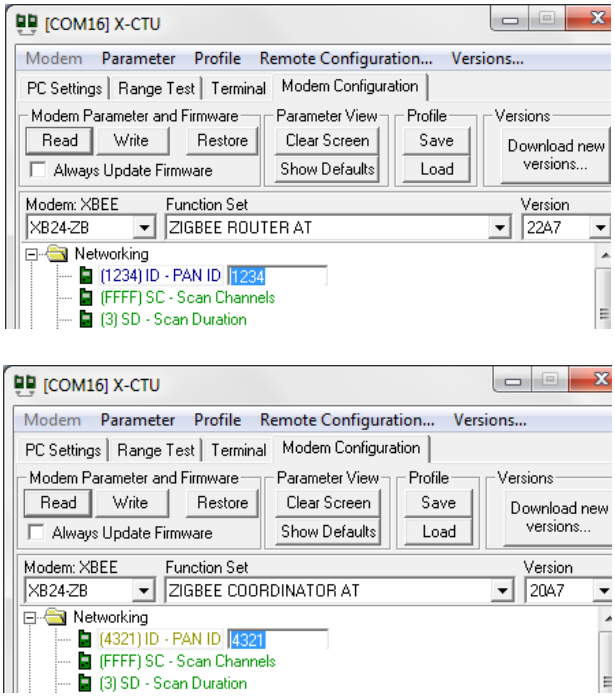


Figura 4.1 Configuración de Módulos Xbee en subredes diferentes.

4.2.ESCENARIO 2: MONITOREO DE TEMPERATURA

En este escenario se presenta el funcionamiento del nodo sensor de temperatura a lo largo de un día. En la figura 4.2 se muestra el nodo sensor de temperatura utilizado para el monitoreo de la temperatura del cultivo.

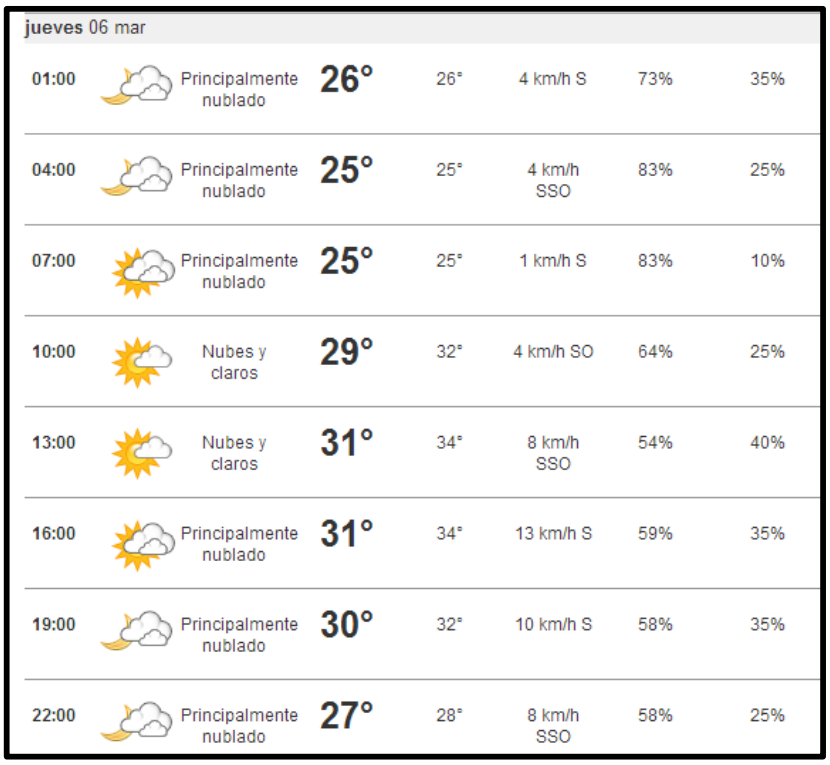
Figura 4.2 Módulo Sensor de Temperatura

En la siguiente tabla se observan las mediciones en tiempo real proporcionadas por el sensor de temperatura tomadas en intervalos de una hora y en la figura 4.3 los datos proporcionados por la aplicación web de pronóstico meteorológico de MSN http://eltiempo.es.msn.com en el transcurso de un día, en base a estos datos realizamos un gráfico de Temperatura vs Tiempo para ambos datos capturados y calculando el área bajo la curva de la figura 4.4 obtuvimos el porcentaje de error, obteniendo como resultado un 3,69% que es un rango aceptable de error, ya que el DS18B20 es uno de los sensores de temperatura con mayor precisión en

el mercado, por medio de lo cual podemos concluir que el nodo satisface por completo los requerimientos para el monitoreo de la temperatura del cultivo.

HORA	TEMP APP WEB	TEMP SENSOR
6:00	25	24
7:00	25	24
8:00	25	25
9:00	25	25
10:00	29	28
11:00	29	29
12:00	29	29
13:00	29	29
14:00	31	30
15:00	31	30
16:00	31	31
17:00	31	31
18:00	31	31
19:00	30	30
20:00	30	31
21:00	30	30
22:00	27	27
23:00	27	26

Tabla 4.1 Mediciones de Temperatura en Tiempo Real.



jueves 06 mar

01:00	Principalmente nublado	26°	26°	4 km/h S	73%	35%
04:00	Principalmente nublado	25°	25°	4 km/h SSO	83%	25%
07:00	Principalmente nublado	25°	25°	1 km/h S	83%	10%
10:00	Nubes y claros	29°	32°	4 km/h SO	64%	25%
13:00	Nubes y claros	31°	34°	8 km/h SSO	54%	40%
16:00	Principalmente nublado	31°	34°	13 km/h S	59%	35%
19:00	Principalmente nublado	30°	32°	10 km/h S	58%	35%
22:00	Principalmente nublado	27°	28°	8 km/h SSO	58%	25%

Figura 4.3 Mediciones de Temperatura proporcionados por el pronóstico meteorológico de MSN [16].

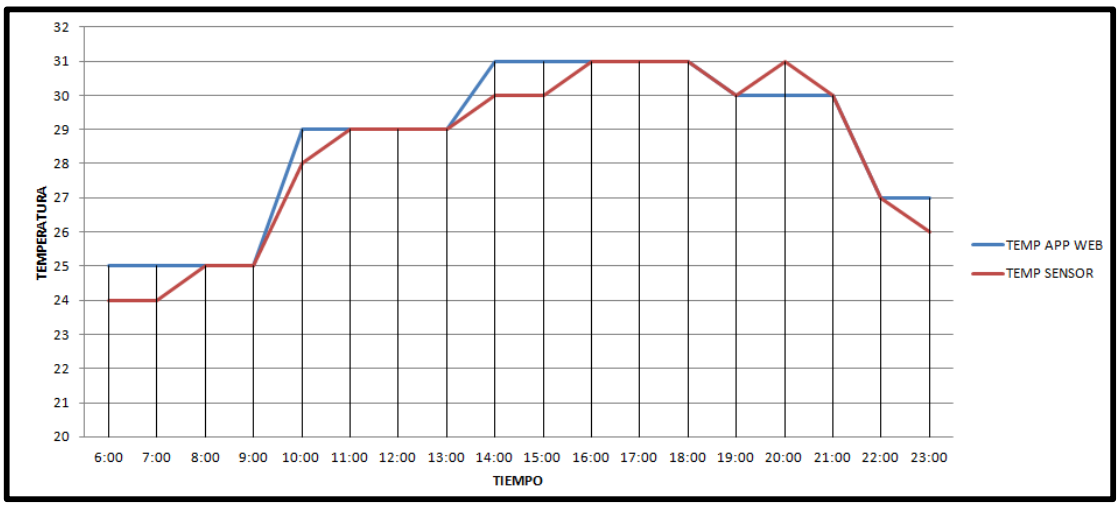


Figura 4.4 Monitoreo de Temperatura vs Tiempo

A continuación se muestra el cálculo realizado para obtener el porcentaje de error de nuestro sensor de Temperatura.

Área bajo la curva experimental (Temperatura del Sensor): 143,5

Área bajo la curva teórico (Temperatura de Aplicación Web): 149

%Error = [(Valor teórico – Valor experimental) / Valor teórico] * 100%

%Error = [(149 – 143,5) / 149] * 100%

%Error = 3,69 %

4.3.ESCENARIO 3: MEDICIÓN DE FRECUENCIA ESTADÍSTICA DE ENVÍO DE DATOS

Para esta prueba se tomaron datos por cada nodo sensor durante 12 horas, la muestra fue tomada de 8:00 am a 08:00 pm, contabilizando para cada nodo el número de eventos por hora, en donde cada evento corresponde a un dato que fue enviado por el sensor a la tarjeta DE2-115.

En la figura 4.5 se muestran los eventos por hora del sensor de temperatura en el periodo de tiempo establecido.

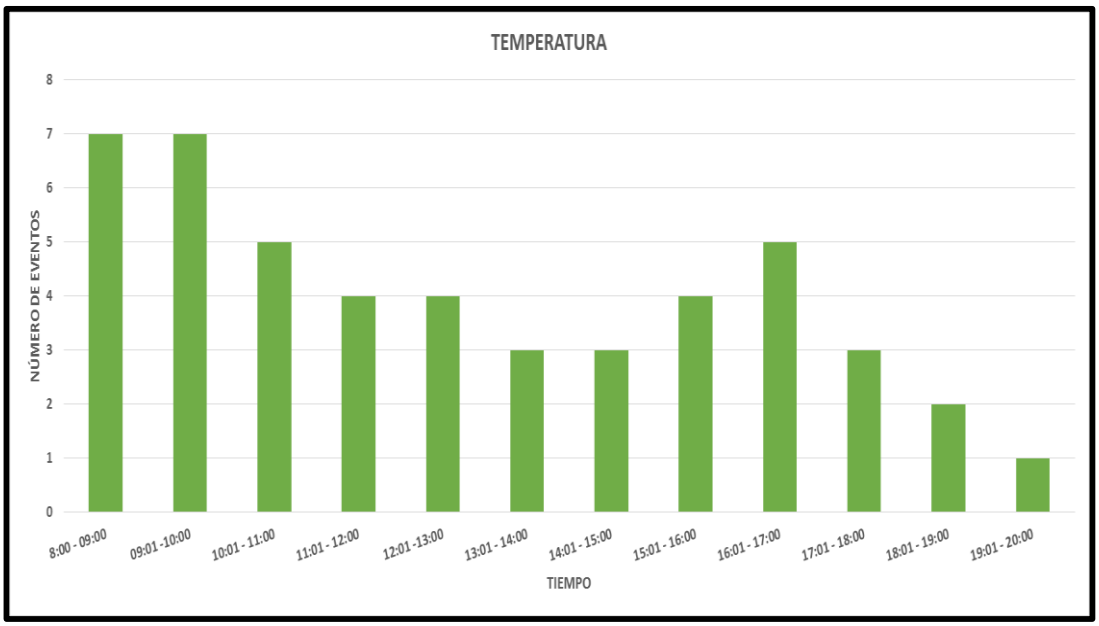


Figura 4.5 Monitoreo de Temperatura

En base a los datos obtenidos concluimos que el sensor de temperatura envía en promedio 4 datos por hora correspondientes a un dato cada 900 segundos (15 minutos).

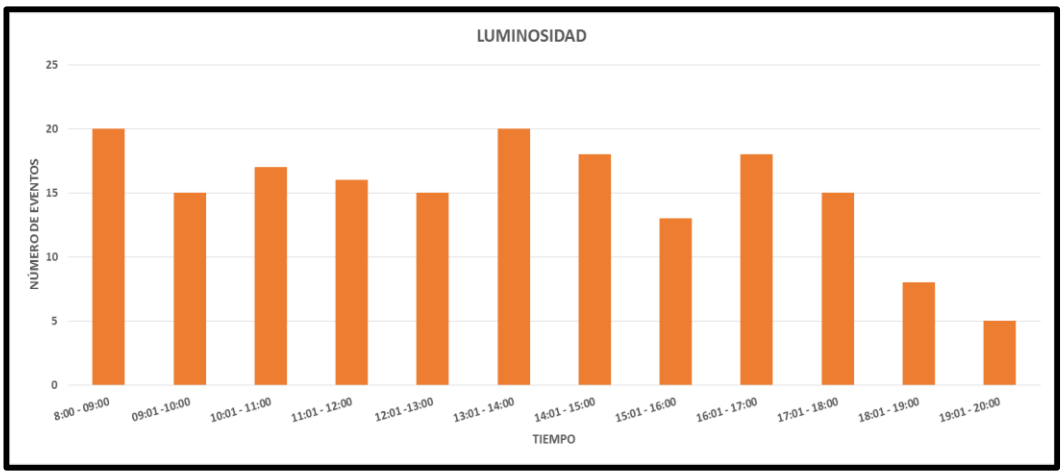


Figura 4.6 Monitoreo de Luminosidad

Así mismo para el caso del sensor de luminosidad de acuerdo a los datos obtenidos y sacando el promedio para estimar la frecuencia de envío del nodo de Luminosidad, concluimos que dicho sensor envía 15 datos por hora correspondientes a un dato cada 240 segundos (4 minutos).

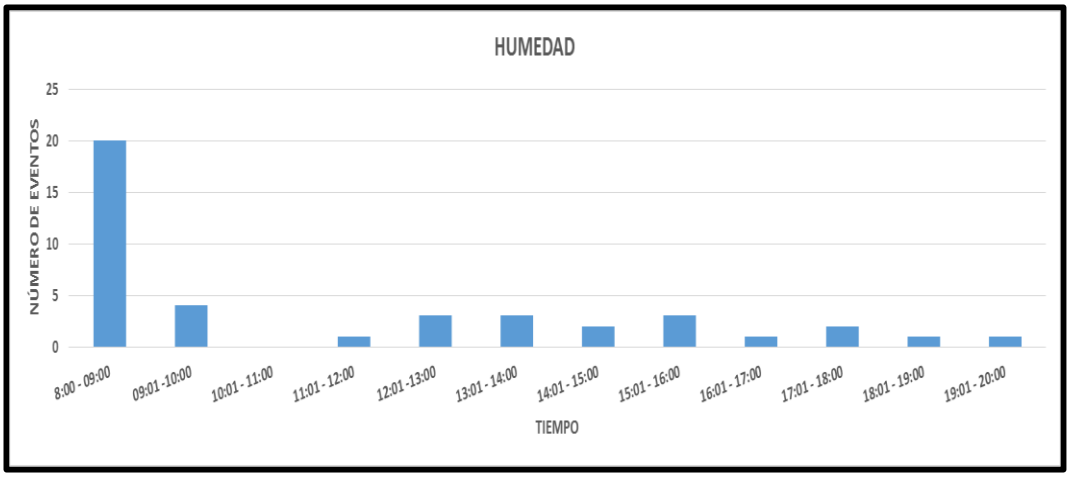


Figura 4.7 Monitoreo de Humedad

Finalmente para el sensor de humedad conforme a los datos recopilados y tomando en consideración el peor caso, es decir el momento en que el sistema de riego está activado, podemos decir que dicho sensor envía 20 datos por hora correspondientes a un dato cada 180 segundos (3 minutos).

Para los sensores de temperatura y humedad se estimó la frecuencia de envío en base a un promedio de los datos recopilados ya que se observó que existe periodicidad en el número de eventos por hora.

4.4.ESCENARIO 4: MEDICIÓN DE FRECUENCIA ESTADÍSTICA DE COLISIÓN DE DATOS

En base a la frecuencia de envío de datos, obtenidos en el escenario 3, en esta prueba se recreó un escenario de los posibles eventos que se suscitarían por cada nodo sensor durante una hora. En la figura 4.8 se observan todos los eventos ocurridos durante los 60 minutos de funcionamiento. Los puntos en los que se repite un evento son las posibles colisiones que podría presentar el sistema, en nuestro caso tenemos una probabilidad del 15% de que dos o más eventos sucedan al mismo tiempo. Sin embargo este porcentaje no representa un problema para el sistema puesto que en el peor caso de colisión, el tiempo de espera máximo para nuestro escenario con 3 nodos, es de 8 segundos y debido a que la máxima frecuencia de envío es de 1 dato cada 180 segundos (3 minutos – Nodo Sensor de Humedad), el tiempo de espera no genera pérdida de datos.

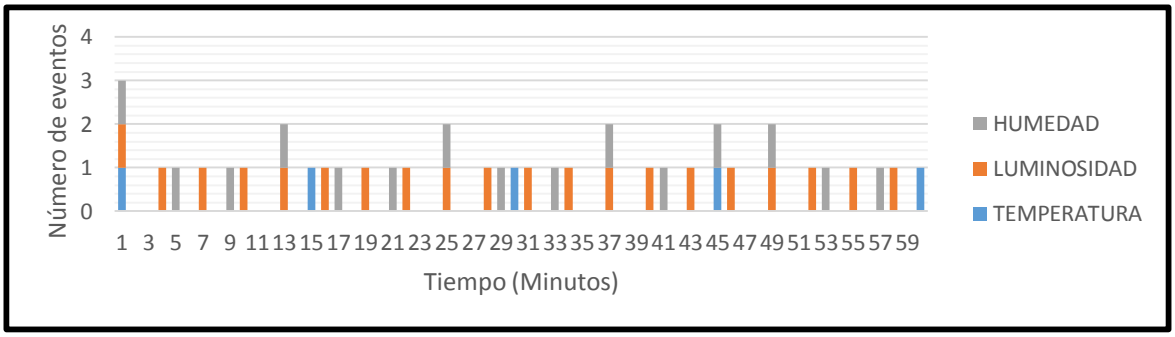


Figura 4.8 Frecuencia Estadística de Colisión de Datos.

4.5.ESCENARIO 5: MEDICIÓN DE CONVERGENCIA DE LA RED.

Para la siguiente prueba se obtuvieron los tiempos de convergencia de la red para todas las combinaciones posibles en un escenario con tres nodos.

Se tomaron tres mediciones de tiempo para cada combinación, obteniendo en cada toma tiempos similares, con lo cual se pudo estimar el tiempo promedio de convergencia de la red.

En la figura 4.9 a, se observa que en el caso de que solamente el nodo de Luminosidad se encuentre encendido, el tiempo de convergencia es en promedio 21.08 segundos, así mismo en la figura 4.9 b el tiempo de convergencia es de 19.74 segundos en promedio cuando sólo se encuentre encendido el sensor de Temperatura y para el caso del sensor de Humedad como se observa en la figura 4.9 c se estima un tiempo promedio de convergencia de 16.66 segundos.

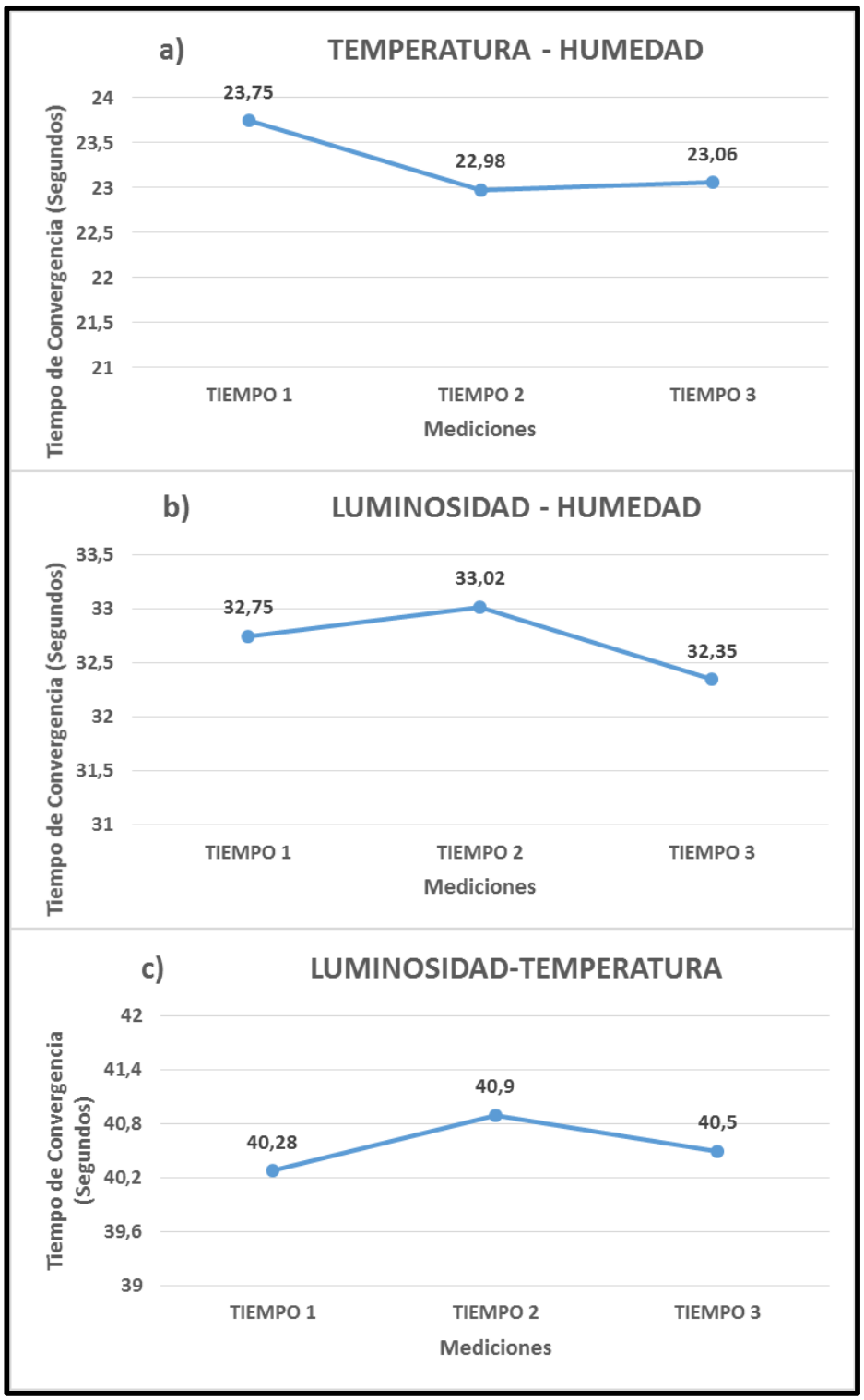


Figura 4.9 Tiempo de convergencia para nodos individuales.

En un escenario con dos nodos obtuvimos los siguientes datos:

NODOS:		TIEMPO 1	TIEMPO 2	TIEMPO 3	T. PROMEDIO
LUM	TEMP	40.28	40.9	40.5	40.56
LUM	HUM	32.75	33.02	32.35	32.71
TEMP	HUM	23.75	22.98	23.06	23.26

Tabla 4.2 Tiempo de Convergencia para Duplas de Nodos.

Para la primera dupla Luminosidad-Temperatura se puede observar claramente que el tiempo de convergencia esta entre 40 y 41 segundos por lo que podemos estimar un promedio 40.56 segundos en converger.

En la segunda dupla Luminosidad-Humedad siguiendo la tendencia se estima que la red converge en 32.71 segundos y para la última dupla de nuestro escenario Temperatura-Humedad el tiempo promedio de convergencia es de 23.26 segundos.

En el último caso cuando se encienden los tres nodos al mismo tiempo, como se observa en la figura 4.10 el tiempo de convergencia es mucho mayor a los dos casos anteriores, obteniendo un tiempo de 52.23 segundos.

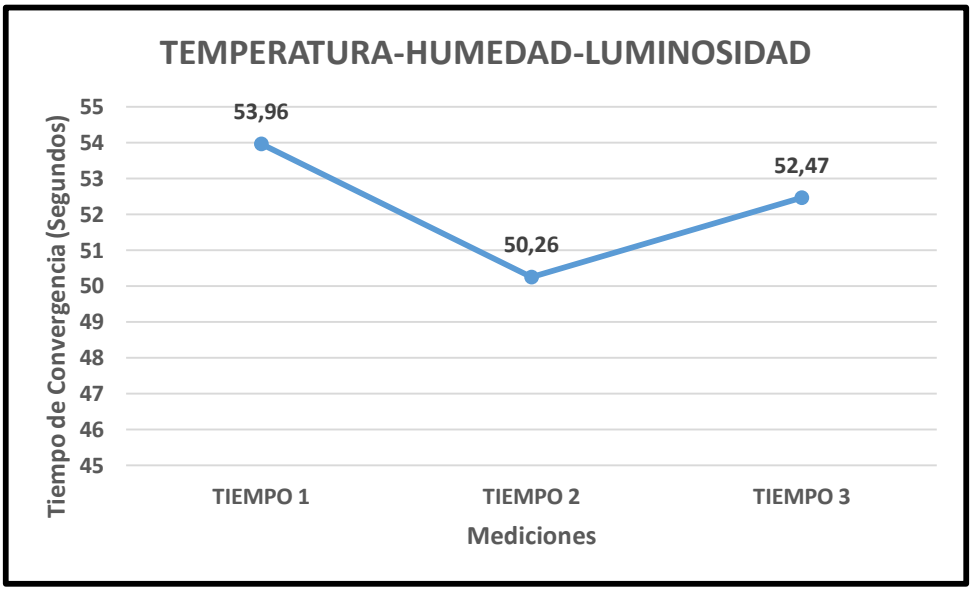


Figura 4.10 Tiempo de Convergencia para tres nodos.

La figura 4.11 representa el aumento en la convergencia de la red en función del número de nodos ingresados.

Para el cálculo de la ecuación de la recta se utilizaron los puntos (1 , 17.5) y (2 , 34.25) de la recta linealizada, obteniendo lo siguiente:

$$Y - 17.5 = \left(\frac{34.25 - 17.5}{2 - 1}\right)(x - 1)$$

$$y - 17.5 = 16.75x - 16.75$$

$$y = 16.75x - 0.75$$

De acuerdo con la ecuación se puede predecir que el aumento en el tiempo de convergencia será de 16.75 segundos por cada nodo ingresado a la red.

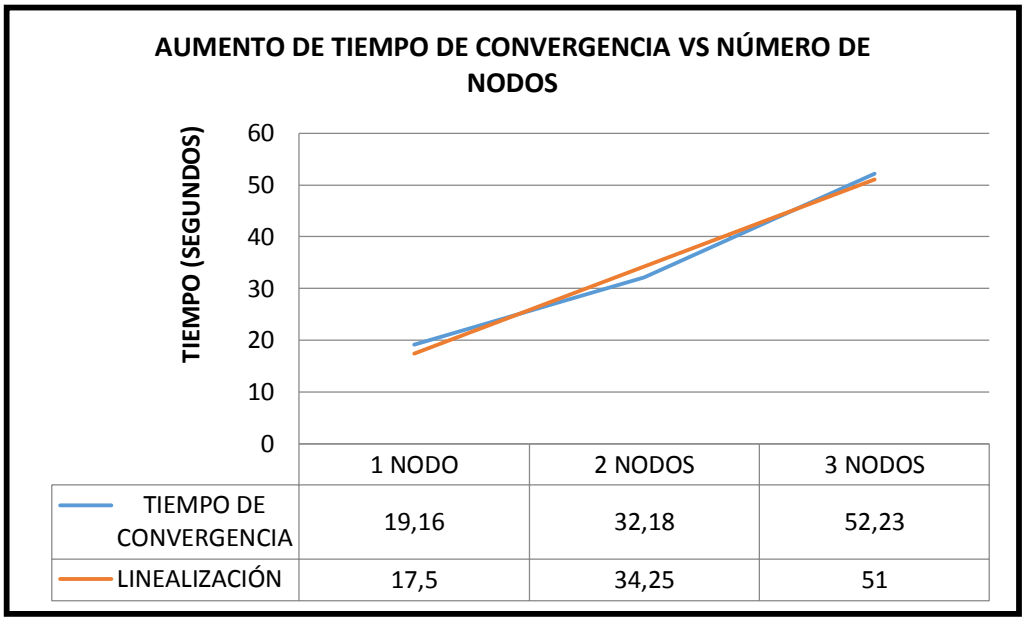


Figura 4.11 Aumento de Tiempo de Convergencia vs Número de Nodos.

4.6.ESCENARIO 6: CAPACIDAD MÁXIMA DEL DATALOGGER.

A continuación se muestra una tabla que indica la cantidad en Bytes por cada dato que se agrega al archivo:

NÚMERO DE DATOS	CANTIDAD EN BYTES
Sin datos	95 Bytes
1 Datos	101 Bytes
2 Datos	107 Bytes
3 Datos	113 Bytes
4 Datos	119 Bytes

Tabla 4.3 Cantidad en Bytes por cada dato agregado al archivo.

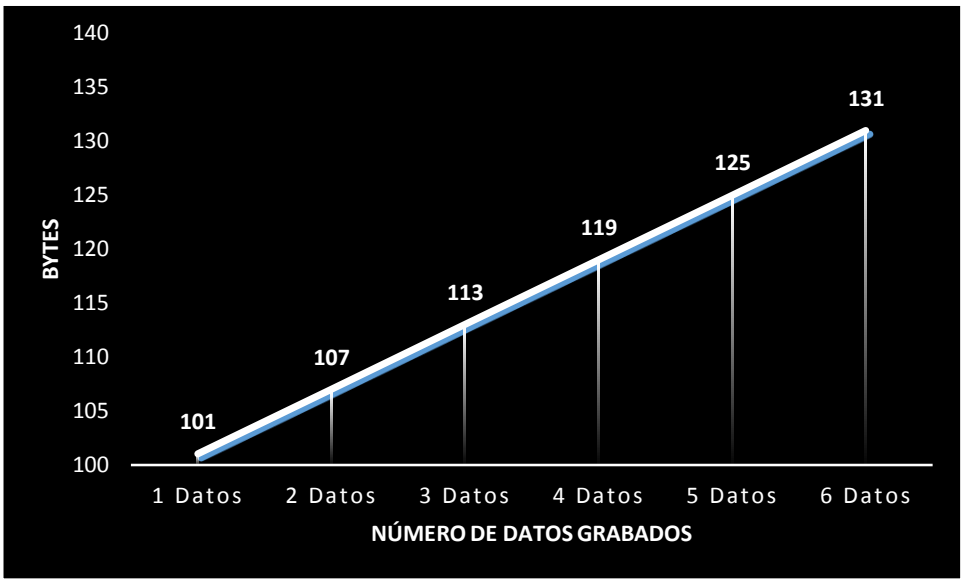


Figura 4.12 Estimación de Capacidad Máxima de Datalogger.

Observando la tendencia de la figura 4.13 se puede estimar que nuestra memoria extraíble con capacidad de 2 GB, en la que en realidad lo máximo a utilizar es 1.83 GB, se puede alojar hasta 305000000 datos. Debido a que la mayoría de cultivos se riegan una vez al día y en base a las pruebas realizadas en donde la cantidad máxima de datos enviados por el nodo sensor de humedad es de 45 por día y cada dato almacenado tiene un peso de 6 Bytes, se estima que el sistema tiene una capacidad de almacenamiento de 6777777 días.

$$Dias\ de\ almacenamiento = \frac{305000000}{45} = 6777777$$

4.7.ESCENARIO 7: ESTIMACIÓN DEL TIEMPO DE DURACIÓN DE LAS BATERÍAS

Cada nodo sensor utiliza el regulador de voltaje Lm7805, el mismo que para entregar una salida de 5 Voltios teóricamente necesita una entrada mínima de 7 Voltios y máxima de 25 Voltios, en la práctica nos pudimos dar cuenta que el valor mínimo de entrada es aproximadamente 7.2 Voltios.

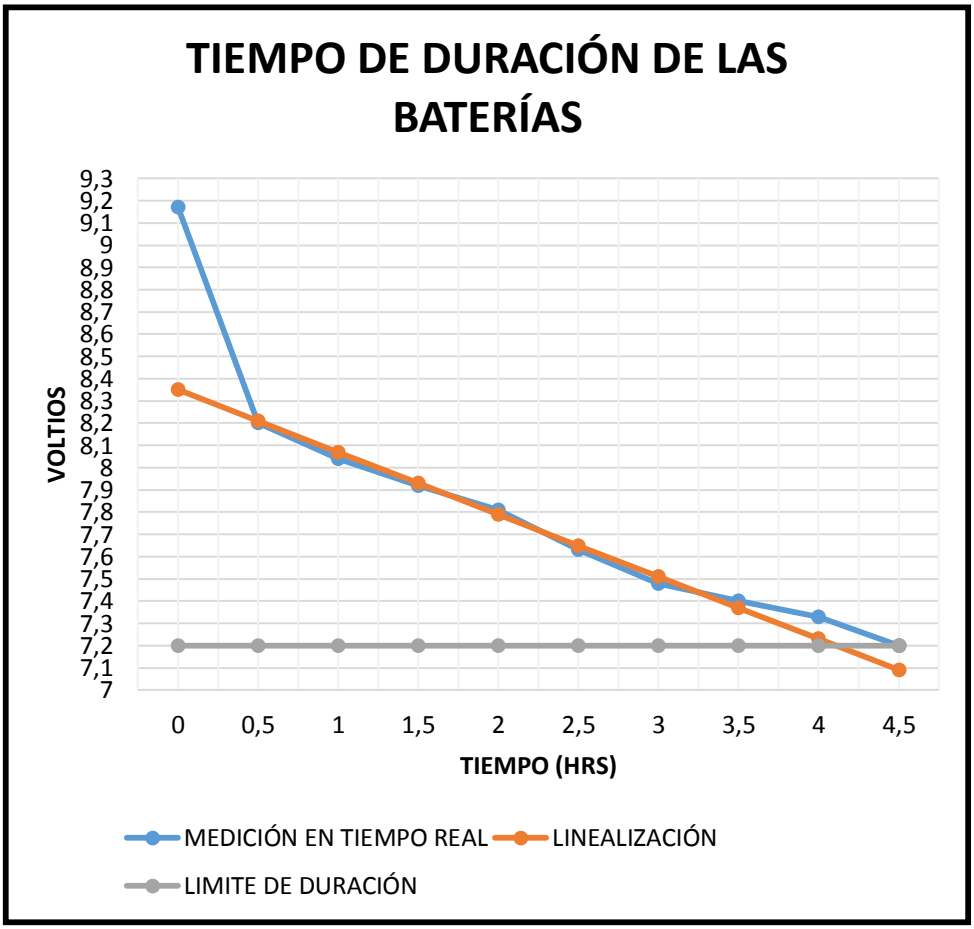


Figura 4.13 Tiempo de Duración de las Baterías.

En la figura 4.11 se muestra el gráfico Voltios vs Tiempo en donde cada punto corresponde al nivel del voltaje de la batería, el mismo que fue tomado cada media hora.

Se tomaron los puntos (0, 8.35) y (1, 8.07) de la recta linealizada y a continuación se muestra el cálculo de la ecuación de dicha recta:

$$y - 8.07 = \left(\frac{8.07 - 8.35}{1 - 0}\right)(x - 1)$$

$$y - 8.07 = -0.28x + 0.28$$

$$y - 8.07 = -0.28x + 0.28$$

$$y = -0.28x + 8.35$$

Dado que cada autómata deja de funcionar cuando el nivel de voltaje es inferior a 7.2 Voltios a continuación se muestra el cálculo del tiempo estimado de la batería:

$$x = \frac{8.35 - 7.2}{0.28}$$

$$x = 4.10\ Horas$$

$$x = 246\ minutos$$

CONCLUSIONES

1. Gracias a los conocimientos adquiridos durante el periodo universitario en programación con lenguaje de alto nivel se pudo generar un sistema para una tarjeta de desarrollo NIOS II que cumple con todas las necesidades definidas para el monitoreo del sistema de riego. El uso del lenguaje C como herramienta para desarrollar un algoritmo para controlar el funcionamiento del proyecto, demuestra la versatilidad que posee dicho lenguaje de programación, lo cual nos permitió implementar un Hardware acorde a los objetivos establecidos.

2. En base a la configuración establecida en cada uno de los nodos sensores y la tarjeta de desarrollo DE2-115 (nodo central), conseguimos una red tipo malla al establecer los parámetros de comunicación en cada uno de los módulos inalámbricos, de manera que todos reciban la información generada por cada nodo sensor y el nodo central, pero solamente se procesa la trama si tiene de cabecera el identificador correspondiente a cada nodo, ya que por ser una red de tipo malla la información se envía en "broadcast", se logró la estabilidad en la comunicación de la red basándonos en el protocolo de acceso compartido al medio CSMA/CD para evitar colisiones.

3. Se implementó un Hardware básico usando la tarjeta DE2-115 de Altera que puede ser usado como guía para futuras mejoras como: detectar cuando uno de los nodos de la red no está operativo, agregar algún otro tipo de nodo como medidor de PH del suelo, entre otras funcionalidades, por lo tanto se puede concluir que se consiguió un sistema adaptable y eficiente con posibilidades de emplear su capacidad y funcionalidad en diseños futuros.

4. Los nodos diseñados para la red inalámbrica satisfacen las expectativas para el sistema de monitoreo de un invernadero, ya que comparados con diseños similares que ya se encuentran en el mercado tienen un margen de error del 3 – 5%.

5. Se lograron cumplir los aspectos sobresalientes de una red inalámbrica de sensores en la implementación ya que el sistema tiene un bajo consumo de energía al trabajar con una batería de 9V, es de fácil instalación e interacción con el usuario, puede ser utilizado en el exterior e interior ya que posee gran movilidad.

6. En la actualidad todas las empresas de cualquier sector productivo se basan en un registro histórico de datos para poder medir los índices de producción y mediante el estudio de los mismos generar mejoras en cuanto a la misma. Nuestro proyecto está dirigido al sector agrícola, sector en el cual los factores más importantes para la producción son la temperatura y humedad de las plantas, para esto creamos un archivo .xls en el cual se encuentran documentados los movimientos a lo largo del día donde se registran los valores, y mediante dichos datos poder generar una curva para facilitar el estudio del monitoreo de la producción del cultivo.

7. Según la ecuación de la recta linealizada la batería se descarga 0.28 Voltios cada media hora y al durar como máximo 246 minutos equivalentes a 4,10 horas no se recomienda usar este tipo de alimentación para el circuito, ya que no se cumpliría uno de los objetivos del sistema que es el monitoreo continuo de los procesos en el invernadero.

8. Para el cálculo del número máximo de nodos que soporta el sistema se tomó en consideración el nodo con la frecuencia más alta de envío, ya que

en base a él se rigen las condiciones del sistema, debido a que de haber una colisión entre todos los sensores de la red, el tiempo de retardo máximo debe ser menor a la frecuencia de envío para que no exista pérdida de datos, en este caso el nodo sensor de humedad (y en general cualquier sensor de humedad que se agregue al sistema), con una frecuencia de envío de 180 segundos es quien limita el número de nodos.

Puesto que la latencia que se agrega a la red es de 4 segundos por cada nodo, el sistema tiene la capacidad de alojar 45 nodos, valor que se obtiene dividiendo 180 (Frecuencia Máxima de Envío) para 4 (Latencia).

9. Debido a que el tiempo máximo de espera es de 180 segundos, el número máximo de nodos que pueden ser encendidos al mismo tiempo es de 10, dato obtenido de igualar el tiempo de convergencia con la frecuencia máxima de envío:

$$y = 16.75x - 0.75$$

$$x = \frac{180 + 0.75}{16.75} \cong 10$$

10. Dado que la cantidad de bytes almacenados por cada dato agregado al archivo es muy baja, tenemos la ventaja de aprovechar los 2GB de espacio disponible de por vida.

RECOMENDACIONES

1. Se recomienda revisar los diagramas esquemáticos de la tarjeta DE2-115 en el momento de conectar algún dispositivo externo para evitar problemas de conexión, daños en los módulos integrados a la tarjeta y sobre todo para evitar algún daño permanente que deje inutilizable por completo la tarjeta de desarrollo DE2-115.

2. Verificar la hoja técnica de cualquier dispositivo externo a usar en el proyecto, como es el caso del módulo inalámbrico Xbee, Microcontrolador PIC16F887, entre otros elementos usados en la implementación de la Red de Sensores Inalámbricos, para evitar daños por algún voltaje mal establecido.

3. Se recomienda usar un software de ayuda para poder visualizar la comunicación entre módulo y la tarjeta a la hora de realizar el proyecto, de esta forma se puede apreciar los errores de programación y cualquier mal configuración de los módulos y evitar la posibilidad de que exista una comunicación errónea entre ellos.

4. Para poder evitar la colisión en el envío y recepción de datos de cada uno de los módulos sensores se los programó de manera tal que exista un carácter de confirmación de envío, recepción y de espera para que se puedan sincronizar y no se pierda la conexión entre ellos y la tarjeta de desarrollo DE2-115.

5. Al momento de configurar los pines de expansión de la tarjeta DE2-115 de Altera es muy importante tener en cuenta la hoja de especificaciones de cada uno de los pines para poder utilizarlos como entrada, salida o alimentación del circuito o elemento a incorporar a la tarjeta.

ANEXOS

ANEXO A

CODIGO FUENTE EN LENGUAJE C

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include "system.h"
#include "altera_up_avalon_character_lcd.h"
#include "altera_up_avalon_character_lcd_regs.h"
#include "nodelist.h"
#include "list.h"
#include "generic.h"

//Definiciones
#define hum_min 11
#define UART_NAME "/dev/UART"

//Estructura Implementada
typedef struct Bomba{
	int numero;
	int humedad;
	char identidad;
}Bomba;

//Funciones Implementadas
void LCD_cursor( int, int );
void LCD_text( char * );
void LCD_cursor_off( void );
int comparar(char c);
Bomba *Bomba_New(int num, int hum, char identidad);
int bomba_search(List *L, char i, int h);
void LCD_clear_screen(alt_up_character_lcd_dev *lcd, int len);
int check_keys(void);
void subir_bajar (alt_up_character_lcd_dev *lcd,int * referencia);
char * convertirIntChar(int num);
void bomba_search_modify(List *L, int h, int hum_new);
```

FUNCIONES Y PROCEDIMIENTOS

```
// FUNCION QUE QUE RECIBE UN CARÁCTER Y LO COMPARA CON UN NUMERO NATURAL Y
RETORNA ENTERO
int comparar(char c)
{
	switch(c)
	{
	//case '0': return 0; break;
	case '1': return 1; break;
	case '2': return 2; break;
	case '3': return 3; break;
	case '4': return 4; break;
	case '5': return 5; break;
	case '6': return 6; break;
	case '7': return 7; break;
	case '8': return 8; break;
	case '9': return 9; break;
	default : return 0;
	}
}

// FUNCION QUE AGREGA UN NODO A LA LISTA CON PARAMETROS PARA IDENTIFICAR
QUE ES UN NODO DE HUMEDAD
Bomba *Bomba_New(int num, int hum, char identidad){
	Bomba * nueva_bomba=(Bomba*)malloc(sizeof(Bomba));
	nueva_bomba->numero=num;
	nueva_bomba->humedad=hum;
	nueva_bomba->identidad=identidad;
	return (nueva_bomba);
}

//FUNCION PARA BUSCAR EL VALOR INGRESADO POR EL USUARIO ASOCIADO A CADA
BOMBA Y COMPARAR
//RETORNA 1 SI EL NODO EXISTE Y LA BOMBA DEBE SER ENCENDIDA
//RETORNA 0 SI EL NODO EXISTE Y LA BOMBA DEBE SER APAGADA
//RETORNA 2 SI EL NODO EXISTE Y LA TIERRA ESTA HUMEDA
//RETORNA 3 SI EL NODO NO EXISTE
int bomba_search(List *L, char i, int h){
	NodeList *iterator;
	Bomba *btemp;
	for(iterator = L->header; iterator!=NULL; iterator = iterator->next
){
		btemp=(Bomba *)nodeListGetCont(iterator);
	    if(btemp->identidad == i){
		if(h>=btemp->humedad)
			return 0;
		else
			if(h<btemp->humedad && h>hum_min)
				return 2;
			else
```

```
                        return 1;
                }
            }
        return 3;
}
/*************************************************************************
*******FUNCION QUE MODIFICA EL PARAMETRO DE HUMEDAD MAXIMA ASOCIADO A CADA
NODO DE **HUMEDAD
*/
void bomba_search_modify(List *L, int h, int hum_new){
      NodeList *iterator;
      Bomba *btemp;
      for(iterator = L->header; iterator!=NULL; iterator = iterator->next
){
            btemp=(Bomba *)nodeListGetCont(iterator);
          if(btemp->numero == h){
            btemp->humedad=hum_new;
            break;
          }
      }
}

/*************************************************************************
******FUNCION QUE CONVIERTO ENTERO A CARACTERES
*/
char * convertirIntChar(int num){
            char numConv[5]="";
            sprintf(numConv, "%d",num);
            return numConv;
      }

/*************************************************************************
*****
 * LCD_clear_screen
 * parametros: alt_up_character_lcd_dev *lcd, int len
 * Procedimiento para borrar la pnatalla del LCD
*************************************************************************
****/
void LCD_clear_screen(alt_up_character_lcd_dev *lcd, int len)
{
      alt_up_character_lcd_set_cursor_pos(lcd, 0, 0);
      alt_up_character_lcd_write(lcd,"                ",len);
      alt_up_character_lcd_set_cursor_pos(lcd, 0 ,1);
      alt_up_character_lcd_write(lcd,"                ",len);
}

/*************************************************************************
*****
 * check_keys
 * Parametros: ninguno
```

```
 * Procedimiento que sirve para validar Botoneras para seleccionar el menu:
Key3 = opcion 1 y Key 2 = opcion 2
 * key 1 = opcion 3
*************************************************************************
************/
int check_keys(void){
	volatile int * KEY_ptr = (int *) 0x10000050;		// pushbutton KEY
address
	int KEY_value;
	do{

		KEY_value = *(KEY_ptr);

	}while((KEY_value == 0));
	return (KEY_value);
}

/*************************************************************************
*****
 * subir_bajar
 * Parametros: alt_up_character_lcd_dev *lcd, int referencia
 * Procedimiento que sirve para incrementar decrementar parámetros a
ingresar
*************************************************************************
****/
void subir_bajar (alt_up_character_lcd_dev *lcd,int * referencia)
{
	LCD_cursor(9,1);
	do{
		LCD_text(convertirIntChar(*referencia));
			if (check_keys()== 4){
				LCD_cursor(9,1);
				LCD_text("       ");
				LCD_cursor(9,1);
				*referencia= *referencia + 1;
				LCD_text(convertirIntChar(*referencia));
			}else if (check_keys()== 8){
				LCD_cursor(9,1);
				LCD_text("       ");
				LCD_cursor(9,1);
				*referencia= *referencia - 1;
				if(*referencia==100)
					*referencia=50;
				LCD_text(convertirIntChar(*referencia));
			}
			usleep(500000);
	}while(check_keys()!=2);					//key 1 (2) en nuestro
proyecto es el boton de enter para grabar el dato

}
```

```
/*********************************************************************
*****
 * LCD_cursor
 * parametros: int x, int y
 * Procedimiendo para posicionar cursor
*********************************************************************
****/
void LCD_cursor(int x, int y)
{
	volatile char * LCD_display_ptr = (char *) 0x10003050;    // 16x2
character display
	char instruction;

	instruction = x;
	if (y != 0) instruction |= 0x40;                    // set bit 6 for
bottom row
	instruction |= 0x80;
		// need to set bit 7 to set the cursor location
	*(LCD_display_ptr) = instruction;                   // write to the
LCD instruction register
}


/*********************************************************************
*****
 * LCD_text
 * Parametros: char * text_ptr
 * Procedimiento para escirbir en pantalla LCD
*********************************************************************
****/
void LCD_text(char * text_ptr)
{
	volatile char * LCD_display_ptr = (char *) 0x10003050;

	while ( *(text_ptr) )
	{
		*(LCD_display_ptr + 1) = *(text_ptr);
		++text_ptr;
	}
}
```

```
/*****************************************************************************
 * LCD_cursor_off
 * parametros: ninguno
 * Procedimiento para apagar el cursor de la pantalla LCD.
*****************************************************************************/
void LCD_cursor_off(void)
{
	volatile char * LCD_display_ptr = (char *) 0x10003050;    // 16x2 character display
	*(LCD_display_ptr) = 0x0C;
                                // turn off the LCD cursor
}


/*Funcion que verifica la SD CARD y el formato*/
void Verificar_SD_Card(alt_up_character_lcd_dev *lcd, int len, int *save)
{
	int connected = 0;
	int i,j;
	char intro[41]="ESCUELA SUPERIOR POLITECNICA DEL LITORAL\0";
	char intro2[29]="RED DE SENSORES INALAMBRICOS\0";
	char intro3[12]="TEMPERATURA\0";
	char intro4[8]="HUMEDAD\0";
	// verifica si existe el HW para leer/escribir memoria SD
	device_reference = alt_up_sd_card_open_dev("/dev/Altera_UP_SD_Card_Avalon_Interface_0");
	if (device_reference != NULL)
	{
		if (*save==0)
		{
		while(1) //Lazo infinito esperando que inserte un SD
		{
			if ((connected == 0) && (alt_up_sd_card_is_Present()))
//Verifica si existe una memoria SD insertada en el slot
			{
			//presenta en la LCD Mensaje
				LCD_clear_screen(lcd,len);
				LCD_cursor (3,0);
				LCD_text ("SD CARD OK\0");
				usleep(500000);
			//Verifica formato de la SD Card
				if (alt_up_sd_card_is_FAT16())
				{
					LCD_clear_screen(lcd,len);
					LCD_cursor (4,0);
					LCD_text ("FORMAT OK\0");
					usleep(1500000);
					// Crea el archivo donde se almacena la Informacion
```

```
                                    base = alt_up_sd_card_fopen("MON.xls",
true);
                                    printf("Base: %d",base);
                                    *save=1;
                                    for(i=0;i<41;i++){  // imprime intro en el
archivo
                                          alt_up_sd_card_write(base,
intro[i]);
                                          usleep(20000);
                                    }
                                    alt_up_sd_card_write(base,0x0D); //Enter
para salto a sgte linea
                                    for(i=0;i<29;i++){

      alt_up_sd_card_write(base,intro2[i]);
                                          usleep(20000);
                                    }
                                    alt_up_sd_card_write(base,0x0D); //Enter
para salto a sgte linea
                                    alt_up_sd_card_write(base,0x0D); //Enter
para salto a sgte linea
                                    for(i=0;i<12;i++){
                                          alt_up_sd_card_write(base,
intro3[i]);
                                          usleep(20000);
                                    }
                                    alt_up_sd_card_write(base,0x09); //Espacio
para grabar en la sgte columna
                                    for(i=0;i<8;i++){
                                          alt_up_sd_card_write(base,
intro4[i]);
                                          usleep(20000);
                                    }
                                    alt_up_sd_card_write(base,0x0D); //Enter
para salto a sgte linea
                                    LCD_cursor(0,1);
                                    LCD_text("TURN OFF SWITCH\0");
                                    usleep(1500000);
                                    break;
                              }
                              else
                              {
                                    LCD_clear_screen(lcd,len);
                                    LCD_cursor (0,0);
                                    LCD_text ("NO SD CARD FAT16\0");
                                    usleep(1500000);
                                    LCD_cursor (0,1);
                                    LCD_text ("1.RETRY 2.EXIT\0");
                                    j=check_keys();
                                    if(j==4)
                                    {
```

```
                                        usleep(1000000);
                                        LCD_clear_screen(lcd,len);
                                        LCD_cursor(0,0);
                                        LCD_text("TURN OFF SWITCH\0");
                                        usleep(1500000);
                                        break; // no esta validado para la
tercera botonera si se aplasta regresa a esperar a la sd card
                                  }
                                        //break;
                                  usleep(1000000);
                                  LCD_clear_screen(lcd,len);
                                  //LimpiarLCD(16, " ERROR FAT16 ","");
                                  //usleep(500000);
                            }
                      connected = 1;
                      }
                      else if ((connected == 0) &&
(alt_up_sd_card_is_Present() == false))
                      {
                            LCD_clear_screen(lcd,len);
                            LCD_cursor (0,0);
                            usleep(1500000);
                            LCD_text ("NO SD CARD\0");
                            usleep(1500000);
                            connected = 0;
                            LCD_cursor (0,1);
                            LCD_text ("1.RETRY 2.EXIT\0");
                            j=check_keys();
                            if(j==4)
                            {
                                  usleep(1000000);
                                  LCD_clear_screen(lcd,len);
                                  LCD_cursor(0,0);
                                  LCD_text("TURN OFF SWITCH\0");
                                  usleep(1500000);
                                  break; // no esta validado para la tercera
botonera si se aplasta regresa a esperar a la sd card
                            }
                            usleep(1000000);
                            LCD_clear_screen(lcd,len);
                            // espera o sale si no encuentra sd card??????
                      }
         }
        }
              else{
                      LCD_clear_screen(lcd,len);
                      LCD_cursor (2,0);
                            // set LCD cursor location to top row
                      LCD_text ("FILE CREATED\0");
                      usleep(5000000);
              }
```

```
	}
	return;
}

/* Graba los datos sensados en la SD Card */
void write_SDcard(short int archivo, int valor)
{
	int primero=0, segundo=0;
	segundo = valor % 10;
	primero = valor / 10;
	alt_up_sd_card_write(archivo,converNum(primero));
	usleep(20000);
	alt_up_sd_card_write(archivo,converNum(segundo));
    usleep(20000);
}

/* Convierte los numeros a su respectivo codigo ASCII
 * para que seean grabados en la SD Card*/
int converNum(int num)
{
	switch(num){
	case 0:	return 0x30; break;
	case 1: return 0x31; break;
	case 2: return 0x32; break;
	case 3: return 0x33; break;
	case 4: return 0x34; break;
	case 5: return 0x35; break;
	case 6: return 0x36; break;
	case 7: return 0x37; break;
	case 8: return 0x38; break;
	case 9: return 0x39; break;
	default: return 0;
	}
}

/* Cierra el archivo creado en la SD Card*/
void Cerrar_Archivo(alt_up_character_lcd_dev *lcd,int len,int *save){
	bool a;
	int i;
	usleep(1500000);
	if (*save==1){
		LCD_clear_screen(lcd,len);
		LCD_cursor (2,0);
			// set LCD cursor location to top row
		LCD_text ("ARE YOU SURE?\0");
		LCD_cursor (2,1);
		LCD_text ("1. YES   2. NO\0");
		VALIDAR:
		usleep(20000);
		i=check_keys();
		if(i==8)
```

```
		{
			a=alt_up_sd_card_fclose(base);
			usleep(2000);
			*save=0;
			LCD_clear_screen(lcd,len);
			LCD_cursor (0,0);
				// set LCD cursor location to top row
			LCD_text ("EXITING FILE\0");
			usleep(400000);
			LCD_cursor (12,0);
				// set LCD cursor location to top row
			LCD_text (".");
			usleep(400000);
			LCD_cursor (13,0);
				// set LCD cursor location to top row
			LCD_text (".");
			usleep(400000);
			LCD_cursor (14,0);
				// set LCD cursor location to top row
			LCD_text (".");
			usleep(400000);
			LCD_cursor (0,1);
				// set LCD cursor location to top row
			LCD_text ("TURN OFF SWITCH\0");
			usleep(2000000);
		}
		else
			if(i==4)
				return;
			else
				goto VALIDAR;
	}usleep(2000000);
	return;
}
```

PROGRAMA PRINCIPAL

```
int main()
{
	FILE *ptr_UART=0;
	char prompt,identificador,senal,uno,cero,tres;
	int decena=0,unidad=0,temp=0,luz=0,humedad=0;
	int grabar=0;
	int numbomb=1;
	prompt='0';
	int i;
	int x,y,z,w;// y= menu , z= submenu , x,w= funcion de devuelve un
numero
	int len=16;
	int hum_t=50,temp_t=25, bomb_t=1;   //variables globales para subir
y bajar valores referenciales en funciones para imprimir
	//int contador=0;
	uno = '1';
	cero = '0';
	tres = '3';
	volatile int *Dip_Switches=(int *) 0x10000040;
	volatile int *UART_ptr=(int *)UART_BASE;
	Bomba *bomb;
	List *Lista_Bombas=listNew();

	alt_up_character_lcd_dev *lcd_dev;
	lcd_dev = alt_up_character_lcd_open_dev ("/dev/Char_LCD_16x2");
//abrir el dispositivo LCD
	ptr_UART = fopen("/dev/UART","r+");
	//Dip_Switches=0x10000040;
	if(ptr_UART==0)
		printf("\nError al abrir %s\n\n",UART_NAME);
	else
	{
while (1)
{

	LCD_clear_screen(lcd_dev, len);
	LCD_cursor (0,0);
		// set LCD cursor location to top row
	LCD_text ("****WELCOME*****\0");
	LCD_cursor (0,1);
		// set LCD cursor location to bottom row
	LCD_text ("*TO WSN PROYECT*\0");
	LCD_cursor_off ();
	// turn off the LCD cursor
	usleep(2000000);
	LCD_clear_screen(lcd_dev,len);
	LCD_cursor (0,0);
	LCD_text ("******MENU******\0");
```

```
	LCD_cursor_off ();
	usleep (2000000);
	MENU:
	usleep(500000);
	LCD_clear_screen(lcd_dev,len);
	LCD_cursor (0,0);
	LCD_text ("[1]ADD  [3]SENSE");
	LCD_cursor (0,1);
	LCD_text ("[2]MODIFY");
	LCD_cursor_off();
	y=check_keys();
	if(y == 8){
	//revisa KEY3
		SUB_MENU:
		usleep(500000);
		LCD_clear_screen(lcd_dev,len);
		LCD_cursor (3,0);
			// set LCD cursor location to top row
		LCD_text ("TURN ON\0");
		LCD_cursor (6,1);
			// set LCD cursor location to top row
		LCD_text ("NODE\0");
		LCD_cursor_off ();
		usleep(2000000);   //tiempo de espera para que el sensor y los
xbee's se reconozcan
		identificador=getc(ptr_UART);   // TOD0 sensor de humedad lo
primero que hace es enviar su identificador
		if((identificador!='%') & (getc(ptr_UART)=='%'))   //se envia
un % como fin de cadena y se envia un 1 como ack
		{// SE PUEDE ELIMINAR LA PARTE DEL FIN DE CADENA YA QUE NADIE
VA A ENVIAR MIENTRAS NO SE LE ORDENE
			//fprintf(ptr_UART,"H"); //1 caracter para que nodos
esperen (Hold)
			LCD_clear_screen(lcd_dev,len);
			LCD_cursor (0,0);
			LCD_text ("HUMIDITY: \0");       //INGRESAR HUMEDAD
MAXIMA - humedad en la que se debe apagar la bomba.
			subir_bajar(lcd_dev,&hum_t);
			LCD_cursor_off();
			bomb=Bomba_New(numbomb,hum_t,identificador);
			listAddNode(Lista_Bombas,nodeListNew(bomb));
			IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,uno);
			numbomb++;
			//printf("OK");
			goto MENU;
		}else
			{
			fprintf(ptr_UART,"0");    //le ordena al sensor que
retome la rutina de emparejamiento
			goto SUB_MENU;
			}
```

```
	}else if (y == 4){
	//revisa KEY2 OPCION 2: MODIFY
		LCD_clear_screen(lcd_dev,len);
		LCD_cursor (0,0);
		LCD_text ("[1] TEMPERATURE\0"); //SE INGRESA NUMERO DE NODO A
MODIFICAR
		LCD_cursor (0,1);
		LCD_text ("[2] HUMIDITY\0");
		LCD_cursor_off();
		usleep(500000);
		z=check_keys();
			if(z==8){
				LCD_clear_screen(lcd_dev,len);
				LCD_cursor (0,0);
				LCD_text ("  TEMPERATURE:  \0");
				subir_bajar(lcd_dev,&temp_t);
				LCD_cursor_off();
				goto MENU;
			}else if(z==4){
				LCD_clear_screen(lcd_dev,len);
				LCD_cursor (0,0);
				LCD_text ("ENTER NODE:\0");
				subir_bajar(lcd_dev,&bomb_t);
				//bomba_search_modify(Lista_Bombas,bomb_t,hum_t);
				LCD_clear_screen(lcd_dev,len);
				LCD_cursor (0,0);
				LCD_text ("    HUMIDITY:   \0");
				subir_bajar(lcd_dev,&hum_t);
				LCD_cursor_off();
				bomba_search_modify(Lista_Bombas,bomb_t,hum_t);
				goto MENU;
			}
		}else if(y==2){
/*Pantalla*/			PANTALLA:
			LCD_clear_screen(lcd_dev,len);
			LCD_cursor (0,0);
			LCD_text ("TEMP:    LUM:  %\0");
			LCD_cursor (0,1);
			LCD_text ("HUMIDITY:   %\0");
/*sensar*/		SENSAR:
			prompt='Z';
			senal='Z';

			while(1) //espera que algun nodo de la red este listo
			{
				//char T='H';
				IOWR_ALTERA_AVALON_UART_CONTROL (UART_BASE, 0);
				IOWR_ALTERA_AVALON_UART_STATUS (UART_BASE, 0);
				if(*Dip_Switches!=0)
				{
					usleep(200000);
```

```
                                    //sub menu con opciones para grabar, dejar
de grabar y regresar a sensar
                                    LCD_clear_screen(lcd_dev,len);
                                    LCD_cursor (0,0);
                                    LCD_text ("1.SAVE  2.STOP\0");
                                    LCD_cursor (0,1);
                                    LCD_text ("3.MAIN MENU\0");
                                    usleep(500000);
                                    w=check_keys();
                                    switch (w)
                                    {
                                          case 8 :
Verificar_SD_Card(lcd_dev,len,&grabar); break;
                                          case 4 :
Cerrar_Archivo(lcd_dev,len,&grabar);break; // funcion para cerrar archivo
                                          default : goto MENU;
                                    }
                                    goto PANTALLA;
                              }
                              if((*(UART_ptr+2) & 0x80)){
                              senal=getc(ptr_UART);
                              usleep(1000);
                              switch (senal){
                              case 'T':
IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,'V');goto Sensar_Temp; break;//
caracter para que solo uno se comunique
                              case 'A':
IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,'M');goto Sensar_Humedad;break;
                              case 'L':
IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,'N');goto Sensar_Luz;break;
                              default: break;
                                 }
                              }
                        }

                        usleep(50000);
// se espera para enviar la señal de 'espera'

/*SENSAR TEMPERATURA*/    Sensar_Temp:
                              decena=comparar(getc(ptr_UART));
                              unidad=comparar(getc(ptr_UART));
                              prompt=getc(ptr_UART);
                              if(prompt!='%')
                                    goto SENSAR;// si hubo descoordinacion
ignora dato y regresa a esperar otro valor
                              decena=decena*10;
                              temp=decena+unidad;
                              usleep(3000);
                              LCD_cursor (6,0);
                              LCD_text (convertirIntChar(temp)); // aqui se
imprime el %de temperatura
```

```
                        if(temp>=temp_t)    //temperatura ingresada x el
usuario
                        {

      IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,senal);

      IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,uno);
                        }
                        else if(temp<=(temp_t - 1))
                        {

      IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,senal);

      IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,cero);
                        }
                        goto SENSAR;

/*SENSAR LUMINOSIDAD*/    Sensar_Luz:
                        decena=comparar(getc(ptr_UART)); // lum:15 %
                        unidad=comparar(getc(ptr_UART));
                        prompt=getc(ptr_UART);
                        if(prompt!='%')
                              goto SENSAR;
// si hubo descoordinacion ignora dato y regresa a esperar otro valor
                        decena=decena*10;
                        luz=decena+unidad;
                        usleep(3000);
                        if(luz<10)
                        {
                              LCD_cursor (13,0);
                              LCD_text("0");
                              LCD_cursor (14,0);
                              LCD_text (convertirIntChar(luz));
 // aqui se imprime el %de luz cuando es menor a 10
                        }
                        else{
                              LCD_cursor (13,0);
                              LCD_text (convertirIntChar(luz));
// aqui se imprime el %de luz cuando es mayor a 10
                        }
                        if(luz<10)   //temperatura ingresada x el usuario
                        {

      IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,senal);

      IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,uno);
                        }
                        else
                        {

      IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,senal);
```

```
	IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,cero);
                        }
                        goto SENSAR;

/*SENSAR HUMEDAD*/        Sensar_Humedad:
                        decena=comparar(getc(ptr_UART));
                        unidad=comparar(getc(ptr_UART));
                        prompt=getc(ptr_UART);
                        if(prompt!='%')
                                goto SENSAR;
// si hubo descoordinacion ignora dato y regresa a esperar otro valor
                        decena=decena*10;
                        humedad=decena+unidad;
                        usleep(3000);
                        if(humedad<10)
                        {
                                LCD_cursor (9,1);
                                LCD_text("0");
                                LCD_cursor (10,1);
                                LCD_text (convertirIntChar(humedad));
// aqui se imprime el %de humedad
                        }
                        else{
                                LCD_cursor (9,1);
                                LCD_text (convertirIntChar(humedad));
// aqui se imprime el %de humedad
                        }

	x=bomba_search(Lista_Bombas,identificador,humedad);
                        usleep(2000);
                        if(x==0)
                        {

	IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,senal);

	IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,cero);
                        printf("Si existe y deberia apagarse %d\n\n",x);
                        }else if(x==1)
                        {

	IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,senal);

	IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,uno);
                        printf("Si existe y deberia prenderse %d\n\n",x);
                        }else
                                if(x==2){

	IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,senal);

	IOWR_ALTERA_AVALON_UART_TXDATA(UART_BASE,tres);
```

```
                                }
                        printf("NODO %c NO EXISTE\n",identificador);
                        if(grabar==1)
 // grabar es 1 cuando sd card ok y archivo creado correctamente
                        {
                                write_SDcard(base,temp);
// datos de temperatura
                                usleep(20000);
                                alt_up_sd_card_write(base,0x09);
 //Espacio para grabar en la sgte columna
                                usleep(20000);
                                write_SDcard(base,humedad);
// datos de humedad
                                usleep(20000);
                                alt_up_sd_card_write(base,0x0D);
//Enter para salto a sgte linea
                                usleep(20000);
                        }
                        goto SENSAR;
                }
}

fclose(ptr_UART);
}
return 0;
}
```

ANEXO B

CÓDIGO FUENTE DEL SENSOR DE LUMINOSIDAD - HUMEDAD

```
' declaracion de variables
dim text as string[14]
dim VALOR,TEMP_VALOR, TempC AS WORD
DIM PORCENTAJE,lum_temp AS FLOAT
DIM viajero,dato,datotemp,datochar,cont,senal,counter,BANDERA as byte


main:
     OPTION_REG = 0X00
' Registro Oscilador de control
     OSCCON = 0X75 ' Oscilador interno de 8mhz
' Registro PUERTO A
     TRISA = 0X01 ' porta.5 como entrada digital
     PORTA = 0X00
' Registro PUERTO B
     TRISB = 0X00 ' porta.5 como entrada digital
     PORTB = 0X00
' Registro PUERTO C
     TRISB = 0X80
     PORTB = 0X00


     PORTD=0
     TRISD=0
' Selección de registro analógico. 1 analógico, 0 digitales
```

```
 ANSEL = 0X01 ' AN<7:0>
 ANSELH = 0X00 ' AN<13:8>
 TRISC=0X80
 PORTC=0X00
 delay_ms(250)
UART1_Init(9600)
viajero=0
dato=0
lum_temp=0
cont=0
'Esta Sección es sólo para el sensor de humedad
 while(TRUE)
         delay_ms(2000) 'espera hasta q xbee se autentique con al red
         if UART1_Data_Ready=1 THEN
           senal=UART1_Read()
           if senal="1" then
              break
           end if
         end if
    UART1_Write("A")
    UART1_Write("%")
 wend
while (TRUE)

     LEER:
     if UART1_Data_Ready=1 THEN
```

```
    DATO=UART1_Read()
END IF

  select case viajero
   case 0
       if dato="A" then
          viajero=1
          BANDERA = 0

       end if
   case 1
        if dato="1" then
          viajero=2
        else
          if dato="0" then
           viajero=3
          else
            if dato="3" then
             BANDERA=0
            end if
        end if
       end if
   end select

 if viajero=2 then
    viajero=0
```

```
    BANDERA = 0
    Portd.rd1=1
    delay_ms(1000)
 end if
 if viajero=3 then
    viajero=0
    BANDERA = 0
    Portd.rd1=0
    delay_ms(1000)
end if
if viajero=5 then
   viajero=0
'Enviar dato
   'UART1_Write("A") '  MAC
   UART1_Write(text[3])
   UART1_Write(text[4])
   UART1_Write("%")
   'UART1_Write(0x0D)
   datochar="X"
   UART1_Init(9600) ' SE INICIALIZA PARA DESCARTAR EL CARACTER
   'DE ESPERA Y ASI PUEDA LEER LA ORDEN DE LA DE2-115
   DELAY_MS(200)
   goto LEER
end if
 IF BANDERA = 1 THEN
   goto REENVIAR
```

```
    ELSE
    IF BANDERA = 2 THEN
      goto REENVIAR2
      END IF
 END IF
  Luminosidad:
VALOR=Adc_Read(0)
TEMP_VALOR=1023-VALOR
TempC=Adc_Read(0)
PORCENTAJE =(TEMP_VALOR /1023)*100
DELAY_MS(600)
if ((lum_temp-Porcentaje)>3) then
   lum_temp=Porcentaje
   DELAY_MS(300)
   UART1_Init(9600) 'ENCERA UART PARA QUE NO VUELVA A LEER
   'REENVIAR:
   while(1)
         for counter = 0 to 2
          if UART1_Data_Ready=1  then
              datochar=UART1_Read()
              if DATOCHAR="M" then
                  WordToStrWithZeros(PORCENTAJE,text)
                  BANDERA = 1
                  viajero=5
                  goto LEER
              end if
```

```
            delay_ms(4000)
           end if
           UART1_Write("A")
           delay_ms(800)
          next counter
          REENVIAR:
          delay_ms(3000)
   wend
else
if ((Porcentaje-lum_temp)>3) then
    lum_temp=Porcentaje
    DELAY_MS(300)
    UART1_Init(9600) 'ENCERA UART PARA QUE NO VUELVA A LEER
    while(1)

        for counter = 0 to 2
          if UART1_Data_Ready=1  then
              datochar=UART1_Read()
              if DATOCHAR="M" then
                 WordToStrWithZeros(PORCENTAJE,text)
                 BANDERA = 2
                 viajero=5
                 goto LEER
              end if
              delay_ms(4000)
           end if
```

```
                  UART1_Write("A")
                  delay_ms(800)
               next counter
               REENVIAR2:
               delay_ms(3000)
            wend
         end if
         end if
         delay_ms(100)
    wend
end.
```

NOTA:

La diferencia entre el código de Luminosidad y Humedad son los identificadores.

CÓDIGO FUENTE DEL SENSOR DE TEMPERARTURA

```
program Nodo_Temp
' declaracion de variables
dim Raw_temp, TempC, comma as word
dim i, j1, j2, minus,datotemp, temp2,dato,datochar,viajero,dato2,viajero2 as
byte
dim counter,BANDERA      as byte
dim text                   as string[14]
main:
     OPTION_REG = 0X00
' Registro Oscilador de control
     OSCCON = 0X75 ' Oscilador interno de 8mhz
' Registro PUERTO A
     TRISA = 0X00 ' porta.5 como entrada digital
     PORTA = 0X00
' Registro PUERTO B
     TRISB = 0X02 ' porta.5 como entrada digital
     PORTB = 0X00
' Seleccion de registro analogico. 1 analogico, 0 digitales
     ANSEL = 0X00 ' AN<7:0>
     ANSELH = 0X00 ' AN<13:8>
     text = "000.0000" ' formato del texto a mostrar
     TRISC=0X80
     PORTC=0X00
     TRISD=0X00
```

```
 PORTD=0X00
 delay_ms(250)
UART1_Init(9600)
VIAJERO=0
VIAJERO2=0
BANDERA = 2
Temp2=85
while (TRUE)
      LEER:
      if UART1_Data_Ready=1 THEN
         DATO=UART1_Read()
       END IF

       select case viajero
        case 0
            if dato="T" then
               viajero=1
               BANDERA = 0
            end if
        case 1
             if dato="1" then
               viajero=2

             else
               if dato="0" then
                viajero=3
```

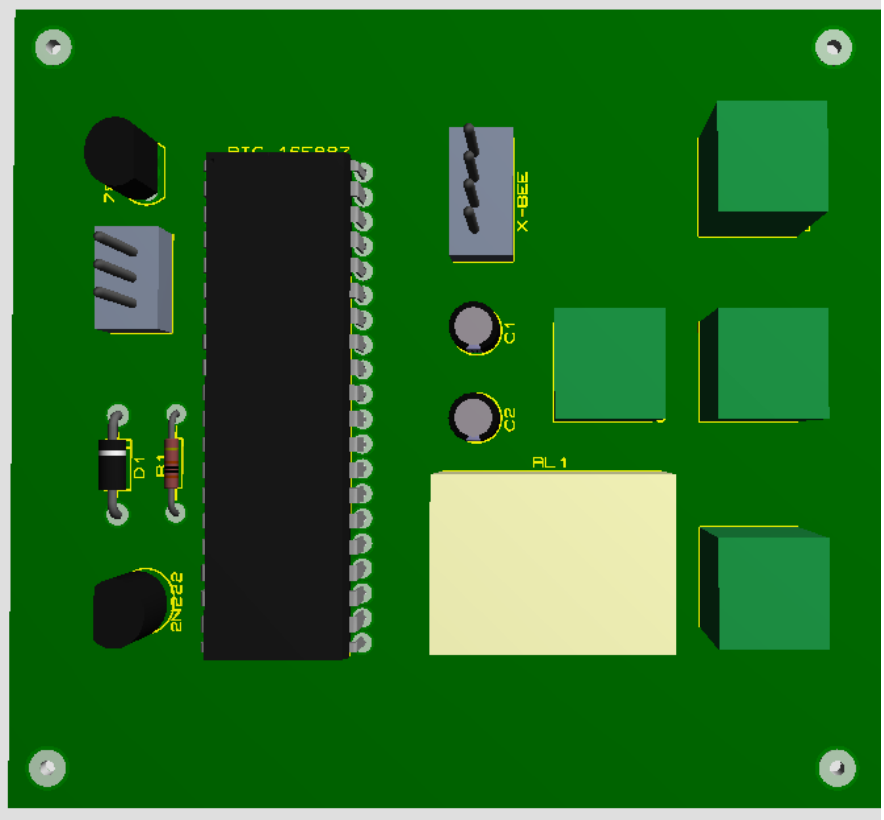
```
          end if
       end if
   end select

 if viajero=2 then
    viajero=0
    BANDERA=0
    Portd.rd1=1
    delay_ms(1000)
 end if
 if viajero=3 then
    viajero=0
    BANDERA=0
    Portd.rd1=0
    delay_ms(1000)
end if



if viajero=5 then
'enviar el dato que tenga
     viajero=0
     UART1_Write(text[3])
     UART1_Write(text[4])
     UART1_Write("%")
     datochar="X"
     UART1_Init(9600) ' SE INICIALIZA PARA DESCARTAR EL CARACTER
```

```
     'DE ESPERA Y ASI PUEDA LEER LA ORDEN DE LA DE2-115
     DELAY_MS(200)
     goto LEER
end if

IF BANDERA = 1 THEN
   goto REENVIAR
END IF
 Temperatura:
 'captura de datos
 if ow_reset(PORTB, 1) = 0 then ' senal de reset onewire
 ow_write(PORTB, 1, 0xCC) ' pase rom
 ow_write(PORTB, 1, 0x44) ' convertir a t
 delay_us(120)
 ow_reset(PORTB, 1)  ' 0 = presente, 1= no presente
 ow_write(PORTB, 1, 0xCC) ' pase rom
 ow_write(PORTB, 1, 0xBE) ' leer a SCRATCHPAD
 j1 = ow_Read(PORTB, 1) ' leer parte baja
 j2 = ow_Read(PORTB, 1) ' leer parte alta
 minus = j2
 minus = minus >> 3
 if minus = 0x1F then ' chequeamos la temperatura (+ o -)
      j2 = not j2
      j1 = not j1
      j1 = j1 + 1
 end if
```

```
 Raw_temp = (j2 << 8) or j1 ' obteniendo la data raw
 TempC = (Raw_temp and $0FF0) >> 4
 comma = (j1 and $0F) * 625
 else
 UART1_Write_Text("X")
 end if
 datotemp=TempC
 Delay_ms(600)

if ((datotemp-Temp2)>=2)then
     temp2=datoTemp
     DELAY_MS(300)
     UART1_Init(9600) 'ENCERA UART PARA QUE NO VUELVA A LEER
     while(1)
        for counter = 0 to 2
          if UART1_Data_Ready=1  then
             datochar=UART1_Read()
             if datochar="V" then
                WordToStrWithZeros(TempC, text)
                BANDERA=1
                viajero=5
                goto LEER
             end if
             Delay_ms(4000)
          end if
          UART1_Write("T")
```

```
                  delay_ms(800)
                next counter
                REENVIAR:
                delay_ms(2300)
            wend
        end if
        delay_ms(100)
    wend
end.
```

ANEXO C



VISTA EN 3D DE LOS NODOS SENSORES

BIBLIOGRAFÍA

[1] Repositorio de la Escuela Superior Politécnica del Litoral, Comunicación a través del Protocolo Zigbee con NIOS II, http://www.dspace.espol.edu.ec/bitstream/123456789/19095/1/Paper%20Jativa-Cabello.pdf, fecha de consulta marzo 2013.

[2] Digi-Key, FPGA Cyclone IV, http://www.digikey.com/product-highlights/mx/es/altera-cyclone-iv-fpga/1886, fecha de consulta marzo 2013.

[3] Jorge Rodríguez Araujo, Estudio del microprocesador NIOS II, http://es.scribd.com/doc/28358833/Estudio-del-microprocesador-NIOS-II, fecha de consulta marzo 2013.

[4] Wikipedia, Field Programmable Gate Array, http://es.wikipedia.org/wiki/Field_Programmable_Gate_Array, fecha de consulta marzo 2013.

[5] Universidad Autónoma de Barcelona, Entorno Gráfico De Configuración Para El Soft-Core Openrisc, http://ddd.uab.cat/pub/trerecpro/2010/hdl_2072_116596/PFC_FelipePintoBuerba.pdf, fecha de consulta marzo 2013.

[6] Wikipedia, Eclipse (software), http://es.wikipedia.org/wiki/Eclipse_(software), fecha de consulta marzo 2013.

[7] Wikipedia, Invernaderos, https://es.wikipedia.org/wiki/Invernadero, fecha de consulta marzo 2013.

[8] Universidad de Cundinamarca, Diseño E Implementación De Un Sistema De Control De Humedad, Temperatura Y Activación De Riego Para El Cultivo De Tomate Bajo Invernadero En La Finca San Sebastián En Arbeláez, http://dc152.4shared.com/doc/-_g6nibc/preview.html, fecha de consulta Abril 2013.

[9] Casadomo, HomePlug y ZigBee, http://www.casadomo.com/noticiasDetalle.aspx?id=7123&c=6, 2005.

[10] TRIPOD, Protocolo RS232, http://juandeg.tripod.com/rs232.htm, fecha de consulta Noviembre 2013

[11] MikroElektronika, Microcontrolador PIC 16F887, http://www.mikroe.com/chapters/view/81/, fecha de consulta noviembre 2013, Capítulo 3.

[12] MikroElektroniKa, Creación del primer proyecto en microC PRO for PIC, http://www.mikroe.com/downloads/get/944/es_1st_project_c_pro_pic_v101.pdf, fecha de consulta noviembre 2013.

[13] Rapidlibrary, Manual de los Xbee, http://technohall.com/2012/12/xctu-herramienta-modulos-xbee/, fecha de consulta noviembre 2013.

[14] 5Hz Electrónica, Sensor de Temperatura Digital, http://www.5hz-electronica.com/sensordetemperaturadigital-ds18b20.aspx, fecha de consulta diciembre 2013.

[15] Hobby Electronics, Arduino – Sensor de Temperatura Digital DS18B20, http://www.hobbytronics.co.uk/ds18b20-arduino, fecha de consulta diciembre 2013.

[16] MSN – Pronóstico Metereológico, http://eltiempo.es.msn.com, fecha de consulta marzo 2014.