



TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE CIUDAD GUZMÁN

TESIS

TEMA:

SISTEMA DE MONITOREO INALÁMBRICO ZIGBEE DE
TEMPERATURA PARA INVERNADERO.

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN INGENIERÍA ELECTRÓNICA

PRESENTA:

ING. RUBEN MEDINA ARIAS

DIRECTOR DE TESIS:

M.C. JAVIER GUTIERREZ TORRES

CIUDAD GUZMÁN JALISCO, MÉXICO, AGOSTO DE 2018



Cd. Guzmán, Jal. a **07/Agosto/2018**

OFICIO No. S/N

ASUNTO : AUTORIZACIÓN DE IMPRESIÓN

ING. RUBEN MEDINA ARIAS
M16290023

En cumplimiento con el documento normativo de las disposiciones para la operación de estudios de posgrado del Tecnológico Nacional de México y con base en la aprobación del Comité Tutorial comisionado para la revisión, la División de Estudios de Posgrado e Investigación le otorga la autorización de impresión de su trabajo de tesis intitulado:

“SITEMA DE MONITOREO INALÁMBRICO ZIGBEE DE TEMPERATURA PARA INVERNADERO”, dirigido por el M. C, JAVIER GUTIERREZ TORRES, desarrollado como requisito parcial para la obtención del grado de Maestro en Ingeniería Electrónica, de acuerdo al plan de estudios MPIEO-2011-13.

Sin otro asunto en particular, quedo de usted.

ATENTAMENTE


DR. HUMBERTO BRACAMONTES DEL TORO
JEFE DE DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN



S.E.P. TecNM
INSTITUTO TECNOLÓGICO
DE CD. GUZMAN
DIVISION DE ESTUDIOS
DE POSGRADO E
INVESTIGACION

C.p. Archivo



Resumen

El objetivo del proyecto es diseñar e implementar una red inalámbrica de sensores basada en el protocolo Zigbee, que proporciona todas las características de un sistema alambrado, sin las limitantes que tienen dichos sistemas. La metodología de ingeniería aplicada es para solucionar el problema de monitoreo de temperatura en un invernadero. En este prototipo se utilizarán 2 Arduino uno "MEGA" y otro "UNO" con 2 XBee 2mW Wire Antenna - Series 2 (Mesh) para la etapa de comunicación, en el Arduino MEGA se montará una Pantalla LCD TFT de 2.4", así como un XBee para la recepción de datos a visualizar, y dentro del invernadero el Arduino UNO con el XBee transmisor conectados a la red de sensores implementada, el protocolo de comunicación es implementado automáticamente por la tecnología Zigbee. EL resultado final es un prototipo implementado que nos permite medir la temperatura continuamente en un invernadero y comunicar de forma remota hacia un panel gráfico, usando una red inalámbrica ZigBee para el monitoreo de dicha variable, se pretende que sirva para prevenir cambios bruscos de temperatura que afectan los cultivos.

Palabra(s) Clave: Zigbee, Arduino UNO, XBee, Mesh, LCD TFT

Abstract

The objective of the project is to design and implement a network of sensors based on the Zigbee protocol, which offers all the characteristics of a wired system, without the limitations of these systems. The engineering methodology applied to the problem of temperature monitoring in a greenhouse. In this prototype we use 2 Arduino one "MEGA" and the other "UNO" with 2 XBee 2mW Wire Antenna - Series 2 (Mesh) for the communication stage, in the Arduino MEGA it will be mounted on a 2.4 "TFT LCD screen, as well As an XBee for the reception of data to be visualized, and inside the greenhouse the Arduino UNO with the XBee transmitter connected to the implemented sensor network, the communication protocol is automatically implemented by the Zigbee technology, the final result is an implemented prototype that allows us to measure the temperature continuously in a greenhouse and communicate with a remote table to a graphic panel, using a ZigBee wireless network for the monitoring of said variable, which means that it is about sudden changes in temperature that affect the crops.

Keywords: Zigbee, Arduino UNO, XBee, Mesh, LCD TFT

DEDICATORIA

Este gran proyecto y oportunidad de finalizar este estudio y formación no hubiera sido posible sin el apoyo de mi familia.

Mis padres María de Jesús Arias Aguayo y Rubén Medina Cobián, por su apoyo incondicional, esfuerzo para darme el estudio y todas mis necesidades.

Mis hermanas Dora Elizabeth Medina Arias y Claudia Medina Arias que sin su ejemplo y guía para seguir adelante hubiese sido mucho más difícil.

A mi asesor de Tesis M.C. Javier Gutiérrez Torres por su apoyo y guía durante la realización de este trabajo.

A todos los profesores de posgrado por sus aportaciones y conocimientos compartidos.

A CONACYT por haberme otorgado el apoyo económico para la realización de mis estudios.

A mis compañeros y amigos que de alguna manera aportaron su conocimiento su experiencia y su apoyo incondicional en este gran trabajo.

Y, por último, pero no menos importante a Dios por poner en mi camino a todas las personas que me ayudaron a seguir adelante y que me dieron la oportunidad de realizar esta gran meta.

Contenido

| Temas | Pág. |
|---|------|
| Resumen..... | i |
| Abstract..... | i |
| Índice de Figuras..... | vi |
| Índice de Tablas..... | vii |
| 1.Generalidades..... | 2 |
| 1.1 Introducción..... | 2 |
| 1.2 Definición del problema..... | 3 |
| 1.3 Problema específico..... | 3 |
| 1.4 Justificación del tema..... | 3 |
| 1.5 Pregunta de investigación..... | 4 |
| 1.6 Objetivos..... | 5 |
| 1.6.1 Objetivos generales..... | 5 |
| 1.6.1.2 Objetivos específicos..... | 5 |
| 1.7 Hipótesis..... | 6 |
| 1.8 Aportación esperada..... | 7 |
| 1.9 Limitaciones..... | 7 |
| 1.10 Consideraciones..... | 7 |
| 2. Fundamentos teóricos..... | 9 |
| 2.1 Variable medida..... | 9 |
| 2.1.1 Temperatura..... | 9 |
| 2.1.2 Manejo de temperaturas..... | 10 |
| 2.2 Comunicación inalámbrica..... | 11 |
| 2.3 Tecnología Zigbee..... | 12 |
| 2.3.1 Áreas de aplicación sobre la tecnología Zigbee:..... | 12 |
| 2.3.2 IEEE 802.15.4 y ZigBee definen diferentes tipos de dispositivos:..... | 13 |

| | |
|---|----|
| 2.3.3 Ventajas y desventajas de Zigbee | 14 |
| 2.4 Modulo XBee | 15 |
| 2.4.1 ¿QUÉ ES XBEE? | 15 |
| 2.4.2 Series o tipos de XBee | 17 |
| 2.4.2.1 XBee Series 1 (También llamados XBee 802.15.4)..... | 17 |
| 2.4.2.2 XBee Znet 2.5 (Formalmente Series 2) | 17 |
| 2.4.2.3 ZB (El actual módulo Series2) | 17 |
| 2.4.2.4. 2B (El incluso más actual módulo Series2)..... | 18 |
| 2.4.2.5. 900MHz | 18 |
| 2.4.2.6 XSC | 18 |
| 2.4.3 Tipos de Antenas | 19 |
| 2.4.3.1 Chip Antenna | 19 |
| 2.4.3.2 Wire Antenna (Whip Antenna) | 19 |
| 2.4.3.3 u.FL Antenna | 20 |
| 2.4.3.4 RPSMA Antenna..... | 21 |
| 2.4.4 Diferencias entre algunos modelos de Xbee | 21 |
| 2.4.4.1 Regular vs Pro | 21 |
| 2.4.4.2. 900 vs 2.4 | 21 |
| 2.4.5 Topologías para módulos XBee..... | 22 |
| 2.4.5.1 Punto a punto | 22 |
| 2.4.5.2 Estrella..... | 22 |
| 2.4.5.3 Malla: | 22 |
| 2.4.5.4 Árbol: | 22 |
| 2.5 Microcontrolador Arduino UNO | 23 |
| 2.5.1 Especificaciones Arduino UNO..... | 24 |
| 2.6 Microcontrolador Arduino Mega 2560..... | 25 |

| | |
|--|----|
| 2.5.1 Especificaciones Arduino Mega 2560 | 26 |
| 2.7 Alimentación para Arduino Mega y UNO | 27 |
| 2.8 Sensor | 28 |
| 2.8.1 Sensor de temperatura DS18B20 | 29 |
| 2.8.1.1 Especificaciones del sensor DS18B20 | 30 |
| 2.9 Pantalla LCD TFT de 2,4" | 31 |
| 2.9.1 Especificaciones TFT de 2,4..... | 31 |
| 2.10 XBee Explorer USB | 32 |
| 2.10.1 Características XBee Explorer USB..... | 33 |
| 2.11 Modulo Shield Expansión XBee para Arduino | 33 |
| 2.11.1 Especificaciones Modulo Shield | 34 |
| 2.12 XCTU | 35 |
| 2.13 IDE Arduino | 36 |
| 3. Desarrollo del sistema de monitoreo inalámbrico | 40 |
| 3.1 Introducción | 40 |
| 3.2 Preparación de módulos Xbee y configuración..... | 40 |
| 3.3 Conexión de Arduino UNO con Xbee emisor..... | 50 |
| 3.4 Conexión sobre red de sensores DS18B20..... | 51 |
| 3.5 Programación del módulo emisor Arduino Uno, Xbee y Sensor DS18B20. | 56 |
| 3.6 Conexión de Arduino MEGA con pantalla TFT | 60 |
| 3.7 Conexión entre Arduino Mega y Xbee | 61 |
| 3.8 Programación del módulo Receptor Arduino MEGA, Xbee y TFT | 62 |
| 3.9 Alimentación de Dispositivos Tanto emisor como Receptor | 75 |
| 4. Resultados | 77 |
| 5. Conclusiones..... | 83 |
| Referencias bibliográficas..... | 84 |

Índice de Figuras

| Figura | Pág. |
|---|------|
| Figura 2.1. Invernadero desde dentro | 9 |
| Figura 2.3 Representación gráfica de red inalámbrica | 11 |
| Figura 2.4 Modulo XBee..... | 16 |
| Figura 2.5 XBee Chip Antenna..... | 19 |
| Figura 2.6 XBee Wire Antenna..... | 20 |
| Figura 2.7 XBee U.FL. RF Connector | 20 |
| Figura 2.8 XBee conector RPSMA..... | 21 |
| Figura 2.9 Tipos de Redes Xbee..... | 23 |
| Figura 2.10 Arduino "UNO"..... | 24 |
| Figura 2.11 Arduino Mega 2560..... | 26 |
| Figura 2.12 Cable USB/ Serie | 28 |
| Figura 2.13 Fuente externa de poder. | 28 |
| Figura 2.14 Sensor DS18B20..... | 30 |
| Figura 2.15 Diagrama Sensor DS18B20..... | 31 |
| Figura 2.16 Pantalla LCD TFT..... | 32 |
| Figura 2.17 XBee Explorer USB..... | 33 |
| Figura 2.18 Modulo Shield Expansión XBee | 35 |
| Figura 2.19 Interfaz XCTU | 36 |
| Figura 2.20 Interfaz Gráfica IDE..... | 38 |
| Figura 3.1 Identificación de Series Numéricas Xbee..... | 40 |
| Figura 3.2 Diagrama de conexión Xbee Explorer USB, PC. | 41 |
| Figura 3.3 Selección de puerto COM. | 42 |
| Figura 3.4 Configuración de parámetros de puerto XCTU. | 43 |
| Figura 3.5 Agregar Dispositivo seleccionado. | 44 |
| Figura 3.6 Modulo agregado. | 45 |
| Figura 3.7 selección de Firmware Coordinador..... | 46 |
| Figura 3.8 Configuración de Valores Networking Coordinador..... | 47 |
| Figura 3.9 Configuración Firmware Router..... | 48 |
| Figura 3.10 Configuración de Valores Networking Router. | 49 |
| Figura 3.11 Arduino y Shield Expansión XBee..... | 50 |

| | |
|---|----|
| Figura 3.12 Verificación del Programa de Direcciones..... | 53 |
| Figura 3.13 Diagrama de conexión Sensores y Arduino | 54 |
| Figura 2.14 Selección de Placa y puerto en IDE | 55 |
| Figura 3.15 Visor de puerto serial. | 56 |
| Figura 3.16 Representación de Conexión Final Emisor | 60 |
| Figura 3.17 Conexión Arduino Mega Pantalla TFT. | 61 |
| Figura 3.18 | 62 |
| Figura 3.19 Carga de Programa en placa Arduino Mega | 75 |
| Figura 4.1 Dispositivos Finales..... | 77 |
| Figura 4.2 Funcionamiento Receptor | 78 |
| Figura 4.3 Funcionamiento Emisor..... | 78 |
| Figura 4.4 Cambios de temperatura 1 | 79 |
| Figura 4.5 Cambios de temperatura 2..... | 79 |
| Figura 4.6 Cambios de temperatura 3..... | 80 |
| Figura 4.7 Cambios de temperatura 4..... | 80 |
| Figura 4.8 Cambios de temperatura 5..... | 81 |
| Figura 4.9 Cambios de temperatura 6..... | 81 |

Índice de Tablas

| Tabla | Pág. |
|--|------|
| Tabla 2.1 Temperaturas críticas de algunos cultivos en Invernadero | 10 |

CAPÍTULO I **GENERALIDADES**

1.Generalidades

En este capítulo se describen brevemente los antecedentes, problemáticas a resolver, justificación, preguntas de investigación, objetivos e hipótesis, así como las limitaciones y alcances sobre los cuales se trabajó en el proyecto y por último lo que se espera lograr con todo lo anteriormente mencionado.

1.1 Introducción

Este proyecto se apoya en el desarrollo de un prototipo de sistema de monitoreo inalámbrico de temperatura, que se aplicó en un invernadero. El sistema funciona con el protocolo de comunicaciones Zigbee el cual permite la comunicación entre nodos tanto emisores como receptores. Sus módulos consumen muy poca energía ya que cuando no se están usando entran en modo de bajo consumo de forma automática, y dado que en la aplicación no manejara un gran volumen de datos, resulta el protocolo que más se adapta a las necesidades del sistema.

El uso y aplicación de Zigbee facilita el proceso de monitoreo de variables ambientales, permite la reubicación de sensores, dispositivos de transmisión, almacenamiento y procesamiento en lugares remotos sin tener que emplear algún tipo de cableado adicional; esta tecnología permite el cambio de lugar sin afectar su desempeño y configuración.

El sistema ayuda al monitoreo de la variable temperatura, ya que es una variable importante en invernaderos, por lo tanto, es una variable altamente influyente dentro de este ambiente. Adicionalmente en una etapa de producción facilitará al usuario final ver el comportamiento de esta variable, y en una etapa posterior poder ser controlada.

El área de aplicación se consideró porque siendo que Ciudad guzmán una región agrícola con muchos sistemas de producción en invernaderos y sus alrededores,

es importante el desarrollo tecnológico para mejorar las condiciones técnicas y optimizar recursos.

1.2 Definición del problema

En los invernaderos existen diversos límites de temperatura ya sean inferiores o superiores, esto se debe a los diferentes cultivos que se tienen en dicho invernadero, una forma de monitorear estas temperaturas es manualmente o con termómetros fijos, el poder tomar una medición a distancia esto nos ayuda a que las plantas no se dañen y el fruto o producto final salga en óptimas condiciones, cuando se tiene una medición exacta o se monitorea dicha variable constantemente antes de tener algún tipo de daño en el plantío, se puede prever y hacer algún cambio climático dentro del invernadero para cualquier desastre siempre y cuando esta variable se mida con exactitud y constante mente en algún lugar de trabajo lejano.

1.3 Problema específico

El problema está en ¿Cómo obtener los datos como procesar dichos datos y como representar dichos datos?, ¿El obtener dichos datos o estudiarlos obtienen beneficios?

1.4 Justificación del tema

Los estudios de monitoreo de temperatura a distancia son muy necesarios en la zona, ya que es una forma eficiente de monitorear la temperatura a distancia sin necesidad de estar presente en el lugar que se desea medir. Lo anterior requiere una red de sensores de temperatura, y en muchas ocasiones la implementación y el cableado de estos resulta costoso, inclusive en tiempo.

CAPITULO I GENERALIDADES

El problema es mayor cuando se pretende realizar alguna medición en un sistema ya implementado, ya que esto podría requerir el recableado y relocalización de los dispositivos de medición.

El uso y aplicación de Zigbee facilita el proceso de monitoreo de variables ambientales.

Las redes actuales se basan en el cableado estructurado que está definido para empresas y oficinas, las cuales no cumplen con los requerimientos del trabajo bajo condiciones ambientales agresivas experimentadas en los invernaderos o similares.

La representación en un panel grafico con limites visuales sobre los invernaderos con cierto tipo de plantas ayudaría al usuario el identificarlas mejor y visualizar la zona de riesgo sobre la cual se tiene que trabajar, para ellos un panel táctil grafico sería la mejor opción.

En conclusiones este Prototipo puede tener muchas ventajas a partir de lo ya implementado y tiene muchos más temas de investigación, a lo largo de la historia todo este tipo de dispositivos tiene mejoras, sobre todo, en procesamiento e imagen, se puede decir que solo son pequeñas migajas para las siguientes generaciones o para la búsqueda de la innovación.

1.5 Pregunta de investigación

En este apartado se exponen las preguntas de investigación realizadas para el inicio de la investigación que ayudaran a determinar los objetivos planeados:

¿Es posible monitorear la temperatura en un invernadero a través de una red inalámbrica de sensores con tecnología Zigbee y representarla?

1.6 Objetivos

En esta sección se describen todos los objetivos relevantes sobre el tema de investigación, los cuales llevarán a realizarlo dicha investigación.

1.6.1 Objetivos generales

Desarrollar y automatizar un prototipo de sistema de monitoreo de Temperatura en invernaderos mediante redes inalámbricas de tipo Zigbee.

1.6.1.2 Objetivos específicos

Este apartado se describe una lista sobre puntos cruciales y específicos para realizar la investigación de una manera ordenada y congruente.

- Diseñar y construir el dispositivo final Zigbee para lectura remota de temperatura en el invernadero.
- Acondicionamiento y filtrado de la señal sobre los sensores seleccionados.
- Adaptar y configurar el dispositivo concentrador (coordinador) para visualización de temperatura y detección de alarmas sobre las lecturas en el invernadero.
- Desarrollar una interfaz gráfica de usuario para visualizar las variables en un panel gráfico.

CAPITULO I GENERALIDADES

- Elaborar un estudio de clasificación y selección de sensores de variables ambientales.

Realizar investigación útil acerca de la tecnología Zigbee sobre el censo de variables de temperatura en ambientes controlados, para poder desarrollar un sistema automatizado basado en microcontroladores de alto desempeño y tecnología, esto es sobre el monitoreo, control de temperatura de forma inalámbrica y remota a través de una red.

Desarrollar un plan de pruebas ya que el objetivo principal del proyecto es la automatización de los componentes electrónicos.

Realizar investigación documental para evaluar las prestaciones de los posibles materiales a utilizar en el proyecto.

Elaboración del protocolo del anteproyecto, para posteriormente presentarlo para su aprobación.

Realizar correcciones a protocolo de anteproyecto y elaborar el documento final.

1.7 Hipótesis

La hipótesis de este tema de investigación es:

Un prototipo de sistema de monitoreo de temperatura en invernaderos mediante las redes inalámbricas con protocolo Zigbee, y la representación gráfica en un panel.

1.8 Aportación esperada

Se pretende poder desarrollar un prototipo de sistema de monitoreo de temperatura en invernadero mediante red inalámbricas de tipo Zigbee que obtenga datos precisos, así como un manejo de información confiable sin necesidad de las limitantes del cableado además de filtros de señales que ayuden a que las pequeñas o grandes interferencias no afecten sobre los datos mostrados.

1.9 Limitaciones

El prototipo se realizará y se probará en instalaciones de la institución, pero el poder probarlo en diferentes invernaderos con condiciones ambientales variables sería lo ideal para tener la certeza de la confiabilidad un funcionamiento del prototipo.

1.10 Consideraciones

Se tiene que analizar a cerca de cual filtro es el más adecuado para la adecuación de la señal, así como, el tipo de red implementada para su buen funcionamiento y mayor alcance.

CAPÍTULO II FUNDAMENTOS TEÓRICOS

2. Fundamentos teóricos

En este capítulo se describen los fundamentos teóricos que ayudan a entender sobre la aplicación del tema de investigación y tener en claro que materiales son óptimos para su desarrollo.

2.1 Variable medida

característica de cada sujeto que puede ser medido de alguna manera. Llamamos “variable” precisamente porque “varía” de sujeto a sujeto. Una variable es un símbolo que permite identificar a un elemento no especificado dentro de un determinado grupo, como lo es la temperatura, luz, humedad, oxígeno etc.

2.1.1 Temperatura

Los invernaderos centran su atención en propiciar las condiciones óptimas para el cultivo. Esto significa controlar y monitorear las condiciones ambientales óptimas para el cultivo seleccionado y sembrado en el interior de estos sistemas. (En la figura 2.1 se puede observar como puede ser un invernadero) [1].



Figura 2.1. Invernadero desde dentro

2.1.2 Manejo de temperaturas

El manejo de la temperatura se basa en la estructura y diseño del invernadero, así como el grado de tecnificación que éste posea. Haciendo referencia al grado de tecnificación, los invernaderos industriales o de gran tamaño son en los cuales, por lo general, se instalan sensores para el monitoreo de las condiciones ambientales internas, ya que al ser grandes extensiones de cultivo el monitoreo por parte de personas no es preciso ni económica mente viable [2].

Asimismo, el grado de tecnificación hace referencia a la cantidad de equipo industrial instalado en el invernadero (sensores, actuadores y controladores), ya que no todos los invernaderos industriales utilizan actuadores para cambiar las condiciones ambientales monitoreadas [3].

En la tabla 2.1 se puede apreciar los niveles máximos y mínimos sobre las temperaturas de algunos cultivos bajo condiciones climáticas de invernadero.

Tabla 2.1 Temperaturas críticas de algunos cultivos en Invernadero

| TEMPERATURAS CRITICAS DE CULTIVOS HORTICOLAS BAJO INVERNADERO | | | | | | | |
|---|-------------|---------|-----|-------------|---------|-----------|---------|
| ETAPAS | GERMINACION | | | CRECIMIENTO | | FLORACION | |
| ESPECIE | MIN | OPT | MAX | NOCHE | DIA | NOCHE | DIA |
| CHILE | 13 | 25 A 30 | 40 | 16 A 18 | 20 A 25 | 18 A 20 | 25 |
| PEPINO | 12 | 30 | 35 | 18 A 22 | 20 A 25 | 18 A 22 | 20 A 25 |
| MELON | 13 | 28 A 30 | 40 | 20 A 24 | 25 A 30 | 18 A 22 | 20 A 23 |
| HABICHUELA | 12 | 15 A 25 | 30 | 16 A 20 | 18 A 30 | 15 A 20 | 20 A 25 |

2.2 Comunicación inalámbrica

La comunicación inalámbrica o comunicación a distancia, es aquella capaz de enviar una cantidad de datos de un punto a otro (emisor y receptor) sin la necesidad de un agente o un hardware que conecte ambos puntos físicamente.

En general, la tecnología inalámbrica utiliza ondas de radiofrecuencia de baja potencia y una banda específica, de uso libre o privada, para transmitir entre dispositivos [4].

En la figura 2.3 se aprecia el funcionamiento de una red inalámbrica.

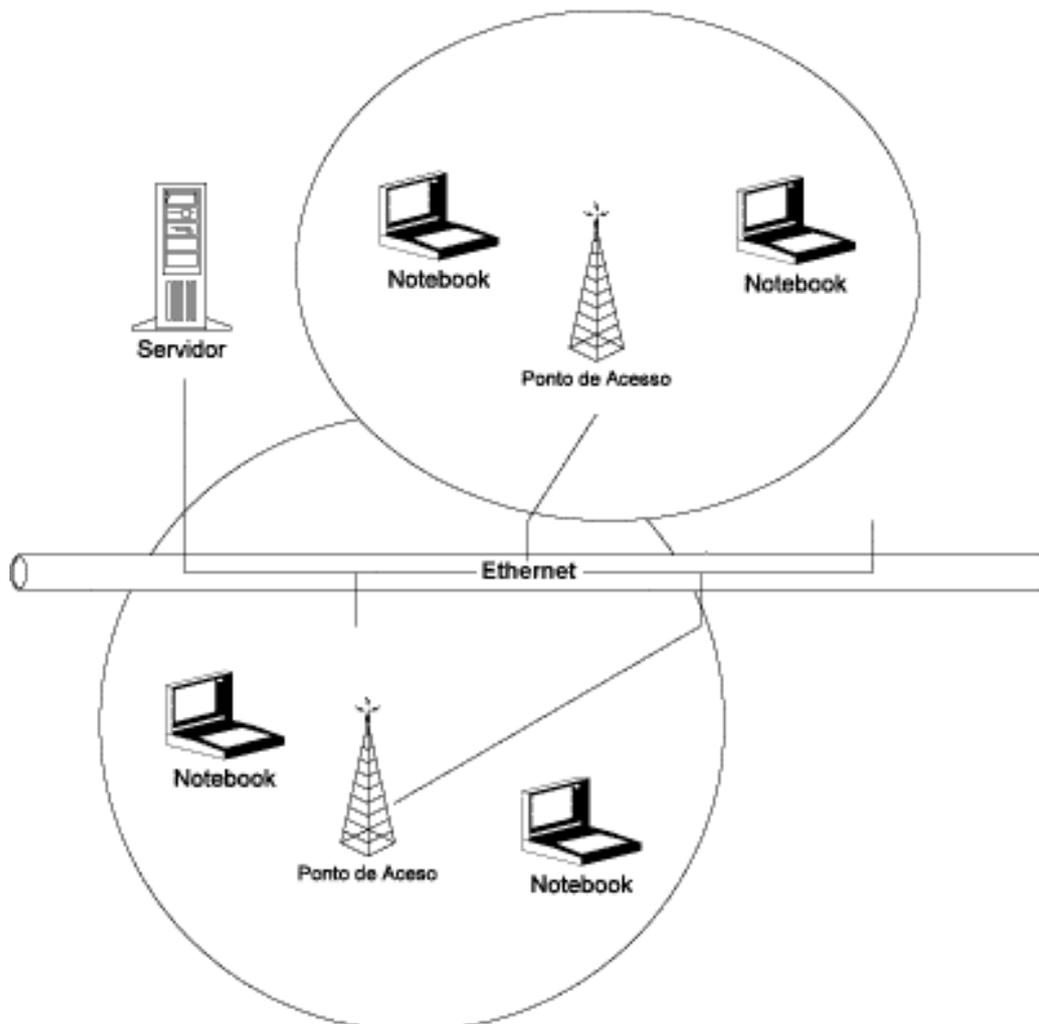


Figura 2.3 Representación gráfica de red inalámbrica

2.3 Tecnología Zigbee

Zigbee se basa en el estándar IEEE 802.15.4, que define el hardware y el software descritos en términos de conexión de red, como la capa física (PHY) y la capa de control de acceso al medio (MAC).

2.3.1 Áreas de aplicación sobre la tecnología Zigbee:

El mercado para las redes Zigbee comprende una amplia variedad de aplicaciones. En la actualidad un gran número de las compañías que forman parte de la Zigbee Alliance se encuentran desarrollando productos que van desde electrodomésticos hasta teléfonos celulares, impulsando el área que más les interesa.

Hay que tener en cuenta que Zigbee está diseñado para aplicaciones que transmiten unos cuantos bytes esporádicamente, que es el caso de una aplicación para automatizar el hogar.

Al usar esta tecnología no habría la necesidad de cablear los interruptores, los cuales podrían ser cambiados de un lugar a otro con plena libertad, pudiendo, por ejemplo, prender o apagar las luces de tu casa a través de Internet o utilizando tu teléfono celular en cualquier momento.

Una de las áreas de aplicación que ha tomado fuerza, es la de los sistemas de medición avanzada, medidores de agua, luz y gas que forman parte de una red con otros dispositivos como displays ubicados dentro de las casas, que pueden monitorear el consumo de energía y no sólo eso, sino que también pueden interactuar con electrodomésticos o cualquier otro sistema eléctrico como bombas de agua o calefacción, con la finalidad de aprovechar mejor la energía.

Zigbee goza de un importante respaldo para la gestión energética y para las soluciones de consumo eficiente por parte de la industria de los servicios públicos;

y por parte de los patrocinadores de las redes energéticas inteligentes en varios países.

Otra área de aplicación prometedora es el rastreo de bienes, también está en la lista la identificación vehicular, nodos ubicados en vehículos que permiten identificar al vehículo a distancia y descargar información que ha recopilado por un periodo de tiempo determinado, monitoreo médico de pacientes y cuidado personal, control de máquinas y herramientas y redes de sensores para el control industrial de plantas de proceso.

Este tipo de escenarios se encuentran al alcance de la tecnología actual. Las anteriores son sólo algunas de las múltiples aplicaciones que se le pueden dar a las redes en cuestión. [5]

2.3.2 IEEE 802.15.4 y ZigBee definen diferentes tipos de dispositivos:

Tipos de dispositivos (según ZigBee):

- Se definen tres tipos distintos de dispositivo Zigbee según su papel en la red:
- Coordinador Zigbee (Zigbee Coordinator, ZC): El tipo de dispositivo más completo. Debe existir uno por red. Sus funciones son las de encargarse de controlar la red y los caminos que deben seguir los dispositivos para conectarse entre ellos.
- Router Zigbee (Zigbee Router, ZR): Interconecta dispositivos separados en la topología de la red, además de ofrecer un nivel de aplicación para la ejecución de código de usuario.
- Dispositivo final (Zigbee End Device, ZED): Posee la funcionalidad necesaria para comunicarse con su nodo padre (el coordinador o un router), pero no puede transmitir información destinada a otros dispositivos. De esta forma, este tipo de

CAPÍTULO II FUNDAMENTOS TEÓRICOS

nodo puede estar dormido la mayor parte del tiempo, aumentando la vida media de sus baterías.

- Un ZED tiene requerimientos mínimos de memoria y es por tanto significativamente más barato. [6]

2.3.3 Ventajas y desventajas de Zigbee

- Ideal para conexiones punto a punto y punto a multipunto.
- Diseñado para el direccionamiento de información y el refrescamiento de la red.
- Opera en la banda libre de ISM 2.4 GHz para conexiones inalámbricas.
- Óptimo para redes de baja tasa de transferencia de datos.
- Alojamiento de 16 bits a 64 bits de dirección extendida.
- Reduce tiempos de espera en el envío y recepción de paquetes.
- Detección de Energía (ED).
- Baja ciclo de trabajo - Proporciona larga duración de la batería.
- Soporte para múltiples topologías de red: Estática, dinámica, estrella y malla.
- Hasta 65.000 nodos en una red.
- 128-bit AES de cifrado - Provee conexiones seguras entre dispositivos.
- Son más baratos y de construcción más sencilla

CAPÍTULO II FUNDAMENTOS TEÓRICOS

- Zigbee tiene un bajo nivel de radiación y, por tanto, se puede utilizar en el sector médico.
- Rango de 10 m a 75m.
- La tasa de transferencia es muy baja.
- Solo manipula textos pequeños comparados con otras tecnologías.
- Zigbee trabaja de manera que no puede ser compatible con Bluetooth en todos sus aspectos porque no llegan a tener las mismas tasas de transferencia, ni la misma capacidad de soporte para nodos.
- Tiene menor cobertura porque pertenece a redes inalámbricas de tipo WPAN. [7]

2.4 Modulo XBee

2.4.1 ¿QUÉ ES XBEE?

Los XBee's son pequeñísimos chips azules capaces de comunicarse de forma inalámbrica unos con otros. Pueden hacer cosas simples, como reemplazar un par de cables en una comunicación serial, lo cual es genial cuando deseas crear, por ejemplo, un vehículo radio controlado.

Pero ¿Cuántos módulos se necesitan? ¿Cuál es la diferencia entre XBee Series 1 y Series 2? ¿Por qué hay tantos tipos de antena para los XBee? ¿Por qué algunos se llaman XBee PRO?, o los que no dicen PRO ¿Son inadecuados? Y lo más importante, ¿Por qué MCI Electronics tiene tantas tarjetas y placas para el XBee y cuál es la que necesito para mi proyecto?

CAPÍTULO II FUNDAMENTOS TEÓRICOS

Existen muchos tipos diferentes de módulos los cuales vamos a repasar, pero una de las ventajas de estos XBee, es que todos, independiente del modelo o serie, tienen los pines similares. Alimentación, tierra y los pines de comunicación (TX/RX) se encuentran en el mismo lugar, haciendo que los chips sean totalmente intercambiables, para la mayoría de las aplicaciones más simples.

De acuerdo con Digi, los módulos XBee (Figura 2.4). Son soluciones integradas que brindan un medio inalámbrico para la interconexión y comunicación entre dispositivos. Estos módulos utilizan el protocolo de red llamado IEEE 802.15.4 para crear redes FAST POINT-TO-MULTIPOINT (punto a multipunto); o para redes PEER-TO-PEER (punto a punto). Fueron diseñados para aplicaciones que requieren de un alto tráfico de datos, baja latencia y una sincronización de comunicación predecible. Por lo que básicamente XBee es propiedad de Digi basado en el protocolo Zigbee. En términos simples, los XBee son módulos inalámbricos fáciles de usar [8].



Figura 2.4 Modulo XBee

2.4.2 Series o tipos de XBee

2.4.2.1 XBee Series 1 (También llamados XBee 802.15.4)

Son la serie más fácil para trabajar, no necesitan ser configurados, pero incluso así se pueden obtener beneficios. Debido a que son fáciles para trabajar. Para comunicaciones Punto-a-Punto, estos módulos trabajan tan bien como los de la Serie 2, pero sin todo el trabajo de preconfiguración previo.

Un módulo Serie 1 no dirá Serie 1 en él, pero tampoco lo dirá uno de la Serie 2. Si no lo dice, entonces tu módulo es un Serie 1. El hardware de las Series 1 y las Series 2/2.5/ZB NO SON COMPATIBLES.

2.4.2.2 XBee Znet 2.5 (Formalmente Series 2)

Los módulos Serie 2 deben ser configurados antes de ser usados. Pueden funcionar en modo Transparente o por medio de comandos API, pero todo esto depende de cual firmware se configure en los módulos. También pueden funcionar en una red mesh, creando unos módulos totalmente configurables e increíbles.

Pero también son más difíciles que usar que los de la Serie 1. No existe una forma en que estos módulos sean compatibles con los de la Serie 1.

Los módulos Znet 2.5 ya no se venden, pero han sido reemplazados con módulos ZB más compatibles.

2.4.2.3 ZB (El actual módulo Series2)

Básicamente es el módulo Znet 2.5, pero con un nuevo firmware. Esto significa que también funcionan en modo transparente o por medio de comandos API. También funcionan en redes mesh. Puedes tomar el nuevo firmware y actualizarlo tú mismo.

CAPÍTULO II FUNDAMENTOS TEÓRICOS

El firmware entre ambos no es compatible (pero es fácilmente intercambiable) por lo que debes elegir cuál firmware deseas usar, y apegarte a éste para crear toda la red.

Estos a menudo son llamados módulos de Serie 2, por lo que, si escuchas a alguien hablar sobre esta serie, probablemente estén hablando de estos módulos. Puede que no sea el término correcto, pero se hace distinción de estos con los módulos de la Serie 1, los cuales son los más populares. Estos módulos no funcionarán de ninguna forma o manera con los de la Serie 1.

2.4.2.4. 2B (El incluso más actual módulo Series2)

Son nuevos módulos que poseen mejoras en el hardware respecto de los de la Serie 2, mejorando por ejemplo el uso de la potencia. Funcionan con el Firmware del módulo ZB, pero debido al cambio de hardware, ya no pueden funcionar con el firmware del módulo Znet 2.5. Por lo que ten cuidado si agregas uno de estos módulos a una red ya existente que utilice módulos Znet 2.5.

2.4.2.5. 900MHz

Técnicamente no es una Serie, pero sí es una familia tal como los otros. Estos módulos pueden funcionar con dos diferentes tipos de firmware, el firmware DigiMesh y el firmware Point-to-Multipoint. Digi (quien fabrica los módulos) actualmente vende ambos módulos, el hardware es el mismo, pero con diferente firmware. MCI Electronics solamente vende la versión Point-to-Multipoint, pero el firmware se puede cambiar fácilmente. Estos módulos son más o menos Plug and Play (llegar y usar) pero por supuesto no se podría tener beneficio de todas las grandes características que en el módulo se pueden configurar.

2.4.2.6 XSC

Son, básicamente, módulos de 900MHz, pero sacrifican velocidad de datos por el alcance. Los módulos 900MHz tienen una velocidad de datos de aproximadamente

156 Kbps (los otros de aproximadamente 250 Kbps), mientras que en los módulos XSC es de alrededor de 10 Kbps. Por otro lado, si colocas una antena de alta ganancia puedes tener un alcance de alrededor de 24 Km y de 9,6 Km con una antena regular. Estos módulos no requieren configuración externa y tienen otras diferencias incluyendo un set de comandos diferente a los anteriores por lo que se recomienda revisar con anterioridad la hoja de comandos.

2.4.3 Tipos de Antenas

2.4.3.1 Chip Antenna

Básicamente es un pequeño chip que actúa como antena. Rápido, sencillo y barato. (Figura 2.5).



Figura 2.5 XBee Chip Antenna

2.4.3.2 Wire Antenna (Whip Antenna)

Es un pequeño cable que sobresale, como se aprecia en la figura 2.6.



Figura 2.6 Xbee Wire Antenna

2.4.3.3 u.FL Antenna

Un conector pequeño para conectar tu propia antena. Esto es perfecto si tienes tu equipo en una caja y deseas la antena afuera de ésta. (Figura 2.7).

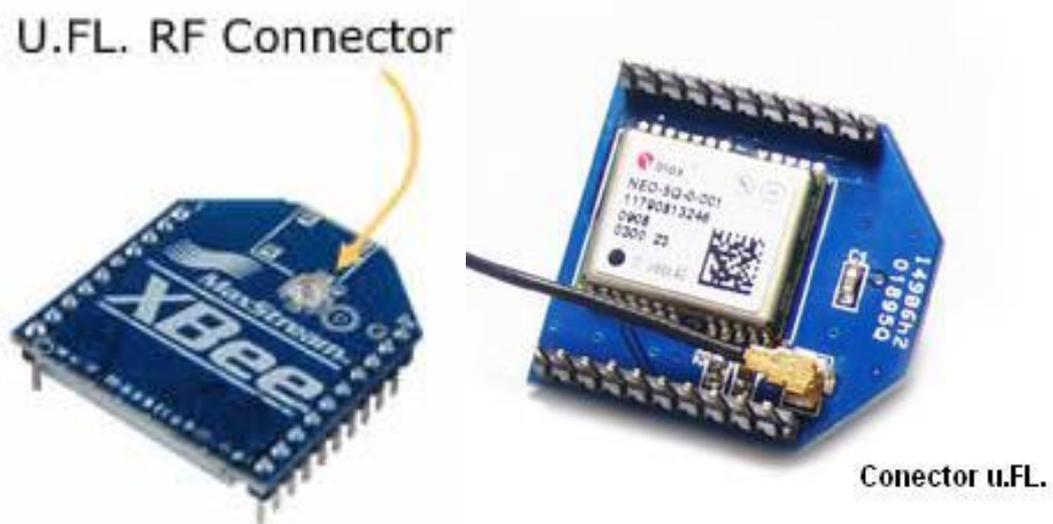


Figura 2.7 Xbee U.FL. RF Connector

2.4.3.4 RPSMA Antenna

Un conector más grande para conectar tu propia antena. Nuevamente, esto es perfecto si tienes tu equipo en una caja y deseas la antena afuera de ésta. (Figura 2.8).



Figura 2.8 XBee conector RPSMA

2.4.4 Diferencias entre algunos modelos de Xbee

2.4.4.1 Regular vs Pro

Hay pocas diferencias entre un XBee regular y un XBee PRO. La diferencia en cuanto a hardware es que el XBee PRO es un poco más largo. Con respecto a comunicación, la versión Pro tiene un mayor alcance (1,6 Km línea vista), pero para ello tiene un mayor consumo de potencia. El criterio para tomar la decisión de cual usar es la distancia que uno requiera comunicar dos XBee. Los dos modelos se pueden mezclar dentro de la misma red.

2.4.4.2. 900 vs 2.4

La mayoría de los módulos XBee operan a 2.4 GHz, pero hay unos pocos que operan a 900 MHz. Básicamente los de 900 MHz pueden llegar muy lejos con una antena de alta ganancia (hasta casi 24 Km). Además, a menor frecuencia, la señal

posee mayor penetración. Otro punto importante es que los módulos de 900 MHz no están permitidos en algunos países (incluso Digi tiene versiones de 868 MHz que sí está permitido en la mayoría de los países). Estas dos versiones no se pueden mezclar en la misma red [8].

2.4.5 Topologías para módulos XBee

Existen diversas topologías de red que se pueden formar con los dispositivos previamente mencionados:

2.4.5.1 Punto a punto

Topología más sencilla, uno de los dispositivos es el coordinador y el otro puede ser un router o un dispositivo final.

2.4.5.2 Estrella

Todos los dispositivos de la red se pueden comunicar con el coordinador, pero no entre sí.

2.4.5.3 Malla:

Se trata de una comunicación punto a punto, pero existiendo restricciones en la intercomunicación de dispositivos.

- Cualquier dispositivo puede comunicarse con otro.
- El coordinador se encarga de la gestión de caminos.

2.4.5.4 Árbol:

- El coordinador establece la red inicial.

CAPÍTULO II FUNDAMENTOS TEÓRICOS

- Los routers forman ramas y retransmiten los mensajes.
- Los dispositivos finales son las hojas del árbol [9].

En la figura 2.9 se puede observar los tipos de redes anteriormente mencionadas.

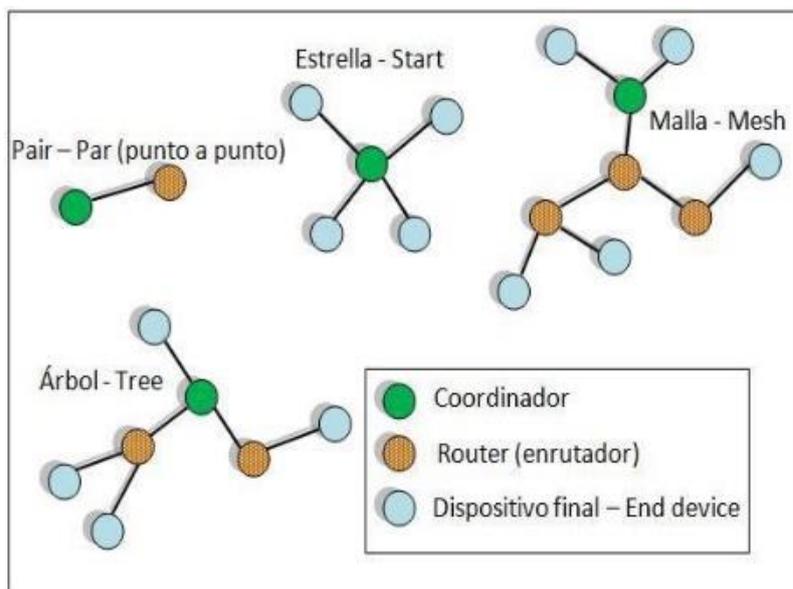


Figura 2.9 Tipos de Redes Xbee

2.5 Microcontrolador Arduino UNO

El Arduino Uno es una placa electrónica basada en el ATmega328. Cuenta con 14 entradas/salidas digitales pines (de los cuales 6 pueden ser utilizados como salidas PWM), 6 entradas analógicas, un oscilador de cristal de 16 MHz, una conexión USB, un conector de alimentación, una cabecera de ICSP, y un botón de reinicio. Contiene todo lo necesario para apoyar al microcontrolador, sólo tiene que conectarlo a un ordenador con un cable USB o a un adaptador AC-DC o la batería para empezar (figura 2.10).

La tarjeta Arduino Uno R3 incluso añade pines SDA y SCL cercanos al AREF. Es más, hay dos nuevos pines cerca del pin RESET. Uno es el IOREF, que permite a los shields adaptarse al voltaje brindado por la tarjeta. El otro pin no se encuentra conectado y está reservado para propósitos futuros. La tarjeta trabaja con todos los shields existentes y podrá adaptarse con los nuevos shields utilizando esos pines adicionales.

El Arduino es una plataforma computacional física open-source basada en una simple tarjeta de I/O y un entorno de desarrollo que implementa el lenguaje Processing/Wiring. El Arduino Uno R3 puede ser utilizado para desarrollar objetos interactivos o puede ser conectado a software de tu computadora (por ejemplo, Flash, Processing, MaxMSP). El IDE open-source puede ser descargado gratuitamente (actualmente para Mac OS X, Windows y Linux).

2.5.1 Especificaciones Arduino UNO

Estas son las especificaciones técnicas de una tarjeta de programación Arduino UNO:

- Microcontrolador ATmega328 (Figura 2.10).
- Voltaje de entrada 7-12V.
- 14 pines digitales de I/O (6 salidas PWM).
- 6 entradas análogas.
- 32k de memoria Flash.
- Reloj de 16MHz de velocidad [10].

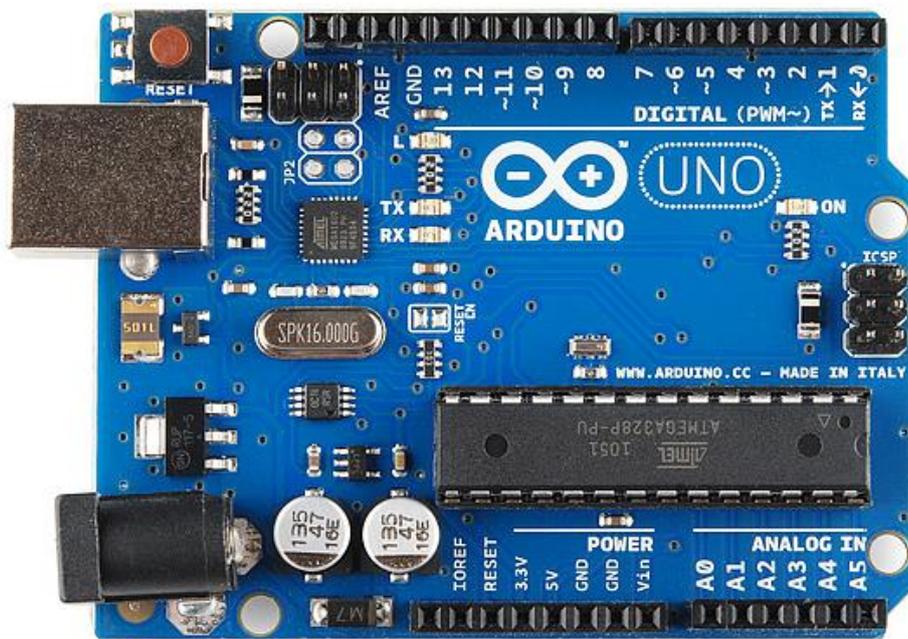


Figura 2.10 Arduino “UNO”

2.6 Microcontrolador Arduino Mega 2560

Arduino Mega es una tarjeta de desarrollo open-source construida con un microcontrolador modelo Atmega2560 que posee pines de entradas y salidas (E/S), analógicas y digitales. Esta tarjeta es programada en un entorno de desarrollo que implementa el lenguaje Processing/Wiring. Arduino puede utilizarse en el desarrollo de objetos interactivos autónomos o puede comunicarse a un PC a través del puerto serial (conversión con USB) utilizando lenguajes como Flash, Processing, MaxMSP, etc. Las posibilidades de realizar desarrollos basados en Arduino tienen como límite la imaginación.

El Arduino Mega tiene 54 pines de entradas/salidas digitales (14 de las cuales pueden ser utilizadas como salidas PWM), 16 entradas análogas, 4 UARTs (puertos serial por hardware), cristal oscilador de 16MHz, conexión USB, jack de alimentación, conector ICSP y botón de reset. Arduino Mega incorpora todo lo necesario para que el microcontrolador trabaje; simplemente conéctalo a tu PC por medio de un cable USB o con una fuente de alimentación externa (9 hasta 12VDC). El Arduino Mega es compatible con la mayoría de los shields diseñados para Arduino Duemilanove, diecimila o UNO.

Esta nueva versión de Arduino Mega 2560 adicionalmente a todas las características de su sucesor utiliza un microcontrolador ATmega8U2 en vez del circuito integrado FTDI. Esto permite mayores velocidades de transmisión por su puerto USB y no requiere drivers para Linux o MAC (archivo inf es necesario para Windows) además ahora cuenta con la capacidad de ser reconocido por el PC como un teclado, mouse, joystick, etc.

2.5.1 Especificaciones Arduino Mega 2560

- Microcontrolador: ATmega2560 (Figura 2.11).
- Voltaje Operativo: 5V
- Voltaje de Entrada: 7-12V
- Voltaje de Entrada(límites): 6-20V
- Pines digitales de Entrada/Salida: 54 (de los cuales 15 proveen salida PWM)
- Pines análogos de entrada: 16
- Corriente DC por cada Pin Entrada/Salida: 40 mA
- Corriente DC entregada en el Pin 3.3V: 50 mA
- Memoria Flash: 256 KB (8KB usados por el bootloader)
- SRAM: 8KB
- EEPROM: 4KB
- Clock Speed: 16 MHz [11].

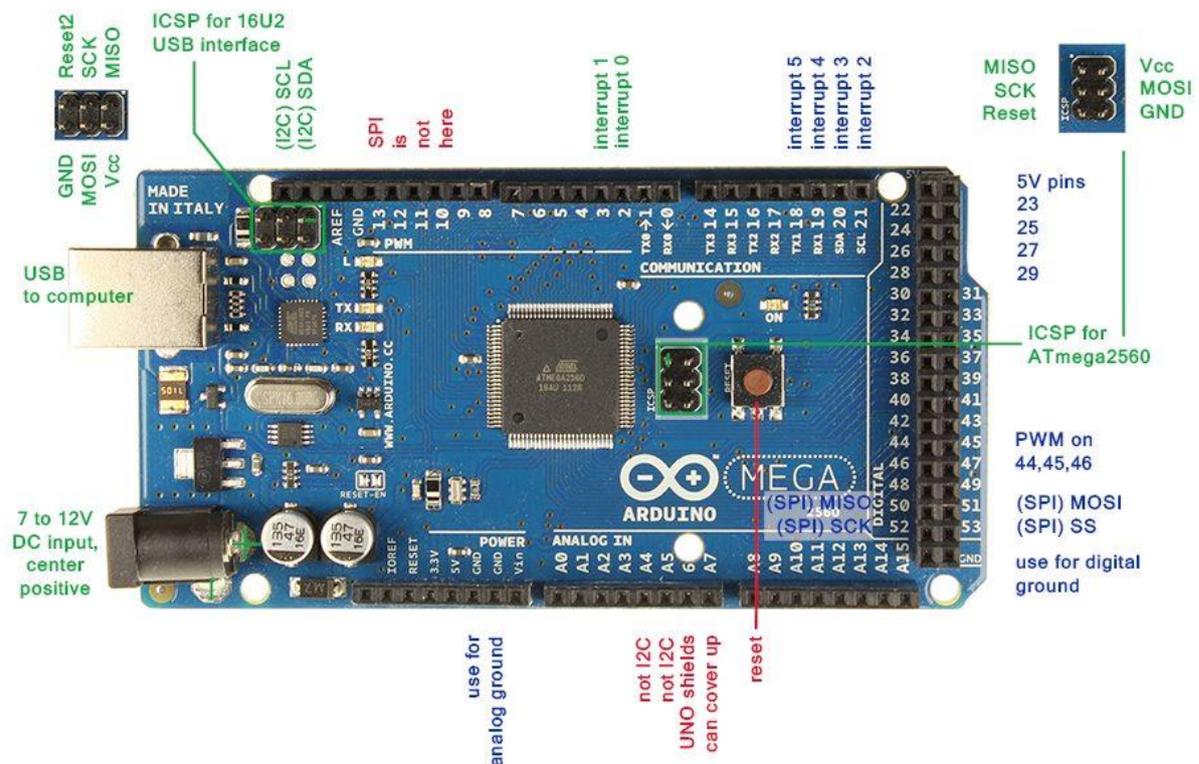


Figura 2.11 Arduino Mega 2560

2.7 Alimentación para Arduino Mega y UNO

Arduino Mega puede ser alimentado mediante el puerto USB o con una fuente externa de poder (Figura 2.13). La alimentación es seleccionada de manera automática.

Cuando se trabaja con una fuente externa de poder se debe utilizar un convertidor AC/DC y regular dicho voltaje en el rango operativo de la placa. De igual manera se puede alimentar el micro mediante el uso de baterías. Preferiblemente el voltaje debe estar en el rango de los 7V hasta los 12V.

Arduino Mega posee algunos pines para la alimentación del circuito aparte del adaptador para la alimentación:

- VIN: A través de este pin es posible proporcionar alimentación a la placa.
- 5V: Podemos obtener un voltaje de 5V y una corriente de 40mA desde este pin.
- 3.3V: Podemos obtener un voltaje de 3.3V y una corriente de 50mA desde este pin.
- GND: El ground (0V) de la placa.

Arduino puede ser programado de una manera muy fácil utilizando el lenguaje propio de Arduino junto con la interfaz Arduino IDE.

La comunicación entre la computadora y Arduino se produce a través del Puerto Serie. Posee un convertidor usb-serie, por lo que sólo se necesita conectar el dispositivo a la computadora utilizando un cable USB como el que utilizan las impresoras (Figura 2.12).



Figura 2.12 Cable USB/ Serie



Figura 2.13 Fuente externa de poder.

2.8 Sensor

Es un dispositivo eléctrico y/o mecánico que convierte magnitudes físicas (luz, magnetismo, presión, temperatura, etc.) en valores medibles de dicha magnitud.

CAPÍTULO II FUNDAMENTOS TEÓRICOS

Esto se realiza en tres fases:

- Un fenómeno físico para medir es captado por un sensor, y muestra en su salida una señal eléctrica dependiente del valor de la variable física.
- La señal eléctrica es modificada por un sistema de acondicionamiento de señal, cuya salida es un voltaje.
- El sensor dispone de una circuitería que transforma y/o amplifica la tensión de salida, la cual pasa a un conversor A/D, conectado a un PC. El convertidor A/D transforma la señal de tensión continua en una señal discreta.

Por ejemplo: existen sensores que se instalan en los vehículos y que detectan cuando la velocidad de desplazamiento supera la permitida; en esos casos, emiten un sonido que alerta al conductor y a los pasajeros.

Otro tipo de sensor muy habitual es aquel que se instala en la puerta de entrada de las viviendas y reacciona ante el movimiento. Si una persona se acerca al sensor, éste emite una señal y se enciende una lámpara. La utilización de estos sensores está vinculada a la seguridad, ya que evitan que alguien aproveche la oscuridad para ocultarse e ingresar en la casa sin ser advertido.

El termómetro también son un tipo de sensor que aprovecha la capacidad del mercurio para reaccionar ante la temperatura y, de este modo, permite detectar si una persona tiene fiebre.

Los sensores, en definitiva, son artefactos que permiten obtener información del entorno e interactuar con ella. Así como los seres humanos apelan a su sistema sensorial para dicha tarea, las máquinas y los robots requieren de sensores para la interacción con el medio en el que se encuentran [12].

2.8.1 Sensor de temperatura DS18B20

Básicamente un sensor de temperatura es un instrumento electrónico capaz de realizar una lectura precisa de una temperatura en un ambiente determinado (Figura 2.14).

2.8.1.1 Especificaciones del sensor DS18B20

Rango de temperatura: -55 a 125°C

Resolución: de 9 a 12 bits

Precisión: $\pm 0.5^\circ\text{C}$ (de -10°C a $+85^\circ\text{C}$)

Tiempo de captura: inferior a 75Ω

Alimentación: 3v a 5.5v

Una de las técnicas utilizadas se denomina RTD (Resistance temperature detector), consiste en detectar los cambios de resistencia de un resistor interno debido a la temperatura (es una medición indirecta) [13].



Figura 2.14 Sensor DS18B20

En la figura 2.15 se puede observar el diagrama eléctrico sobre el funcionamiento y alimentación de el sensor anteriormente descrito:

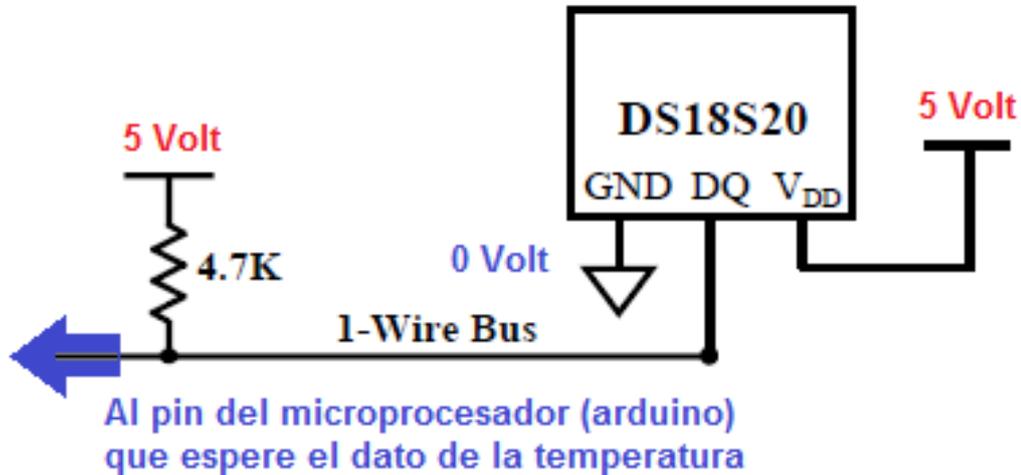


Figura 2.15 Diagrama Sensor DS18B20

2.9 Pantalla LCD TFT de 2,4"

Chip controlador LCD ILI9341, 320x240 pixeles de resolución y 262.000 colores, Panel táctil resistivo de 4 hilos, Compatible con tensión de 5V ya que dispone de regulador de tensión en placa, conexión Shield para Arduino UNO, tamaño: 71 x 52 x 7 mm (L x A x H)

2.9.1 Especificaciones TFT de 2,4

- Pantalla LCD TFT de 2,4" (Figura 2.16).
- Chip controlador LCD ILI9341 (Enlace para descargar el Datasheet ILI9341)
- 320x240 pixeles de resolución y 262.000 colores
- Panel táctil resistivo de 4 hilos
- Compatible con tensión de 5V ya que dispone de regulador de tensión en placa
- Conexión Shield para Arduino UNO
- Tamaño: 71 x 52 x 7 mm (L x A x H) [14].

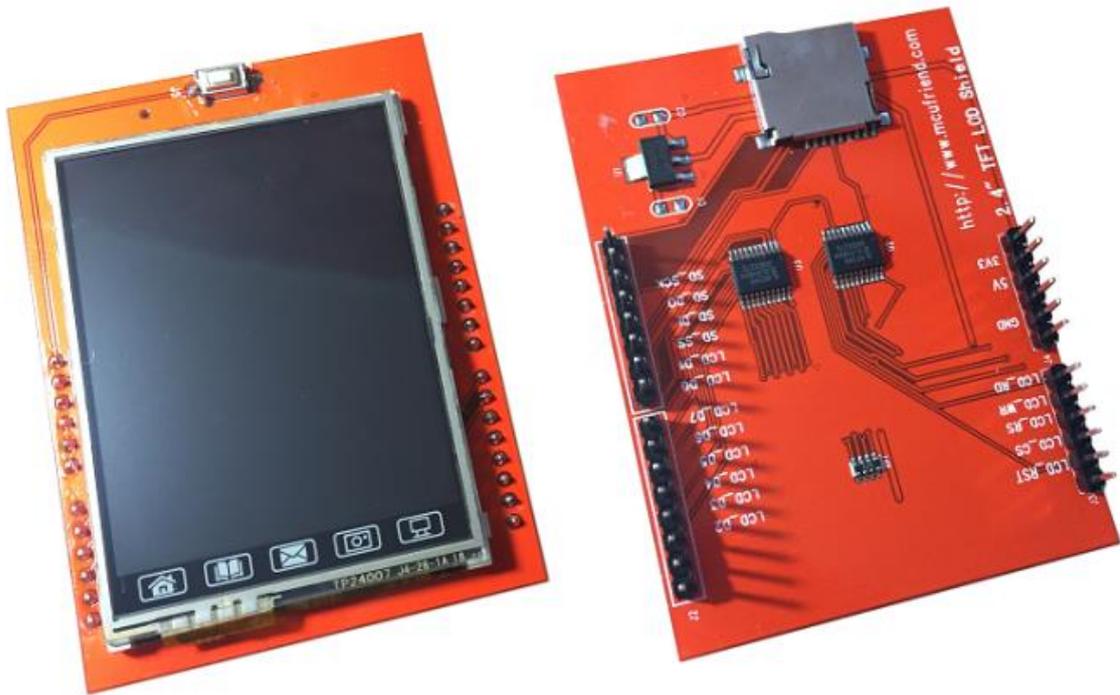


Figura 2.16 Pantalla LCD TFT

2.10 XBee Explorer USB

Es una unidad USB a Serial simple de utilizar diseñada para módulos XBee. Esta unidad funciona con todos los módulos XBee incluyendo la Serie 1 y la Serie 2.5, en versión estándar y Pro. Se puede tener acceso a los pines de seriales y de programación del módulo XBee con sólo colocarlo en la tarjeta y conectar un cable mini USB. Los módulos XBee y el cable USB se venden por separado.

Es un producto altamente útil, ya que además de exponer las señales del puerto serie necesarias para comunicarse con un Arduino y cualquier microcontrolador vía serial (UART), también tiene la capacidad de programar módulos xBee o compatibles para experimentación, configuración y comunicación con el pc.

El Convertidor USB Serial FT232RL es ideal si cuentas con hardware que no posee interfaz serial a bordo (Arduino pro-mini, Arduino, lilypad, etc) o para comunicarte con el bootloader de un microcontrolador cualquiera.

2.10.1 Características XBee Explorer USB

Basada en FT323RL de FTDI para mayor estabilidad en las comunicaciones

Soporta programación y comunicación de módulos inalámbricos XBee y compatibles.

Puede funcionar como una interfaz PC- Xbee

Zócalo para Xbee incluido en la tarjeta

Pines en 90° para comunicación serial entre microcontroladores con UART [15].

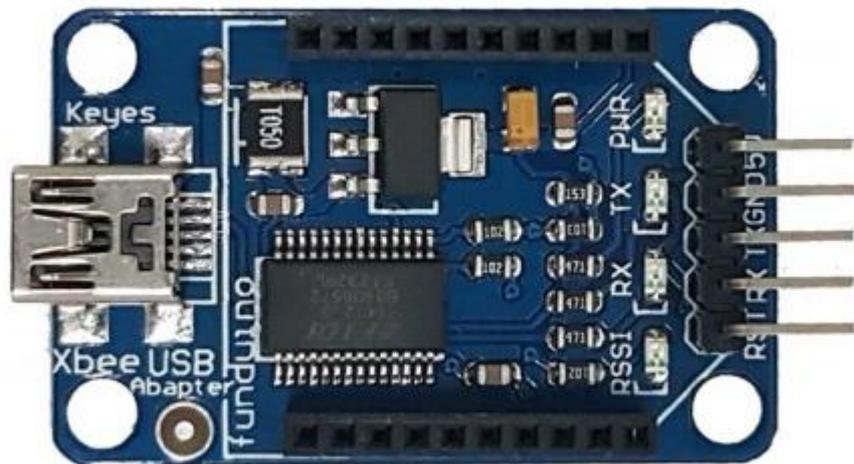


Figura 2.17 XBee Explorer USB

2.11 Modulo Shield Expansión XBee para Arduino

El módulo es fácil de usar y de bajo consumo de energía y el suministro de datos críticos entre dispositivos de transmisión fiable. A medida que el diseño innovador,

XBee-PRO puede estar en el rango de 2-3 veces más allá de los módulos ZigBee estándar. Módulos XBee-PRO funcionan en la banda de frecuencia de 2,4 GHz ISM. El módulo Zigbee XBee de MaxStream (1 mW) es compatible pines.

El módulo Xbee es ampliamente utilizado en los Estados Unidos, Canadá, Australia, Israel y Europa. El establecimiento de la comunicación de RF no requiere ninguna configuración y la configuración por defecto del módulo soporta una amplia gama de aplicaciones de sistemas de datos. También puede utilizar un comando AT simple para la configuración avanzada. Un desarrollador de OEM es ahora paquete de desarrollo de código XBee. Es autodesarrollado en colaboración con el código de módulo RF MaxStream ZigBee / 802.15.4.

2.11.1 Especificaciones Modulo Shield

- Soporta Xbee (Xbee pro)
- Soporta BTBee (HC-06)
- Breakout SPI, IIC, UART BUS
- Botón de Reset
- Tamaño: 60 * 62 * 20 mm
- Placas Soportadas:
- Arduino Uno
- Arduino Mega

Este Shield (una placa de expansión sin módulo XBee) es una solución compatible diseñada para satisfacer redes de sensores inalámbricos de bajo consumo y bajo costo con necesidades especiales. El módulo es fácil de usar, bajo consumo de energía, y la provisión de datos críticos entre dispositivos de transmisión confiable. Como el diseño innovador, XBee-PRO puede estar en la gama 2-3 veces más allá de los módulos estándar de ZigBee. Los módulos XBee-PRO funcionan en la banda de frecuencia ISM 2.4 GHz. El módulo Zigbee XBee (1 mW) de MaxStream es compatible con pin. (figura 2.18) [16].

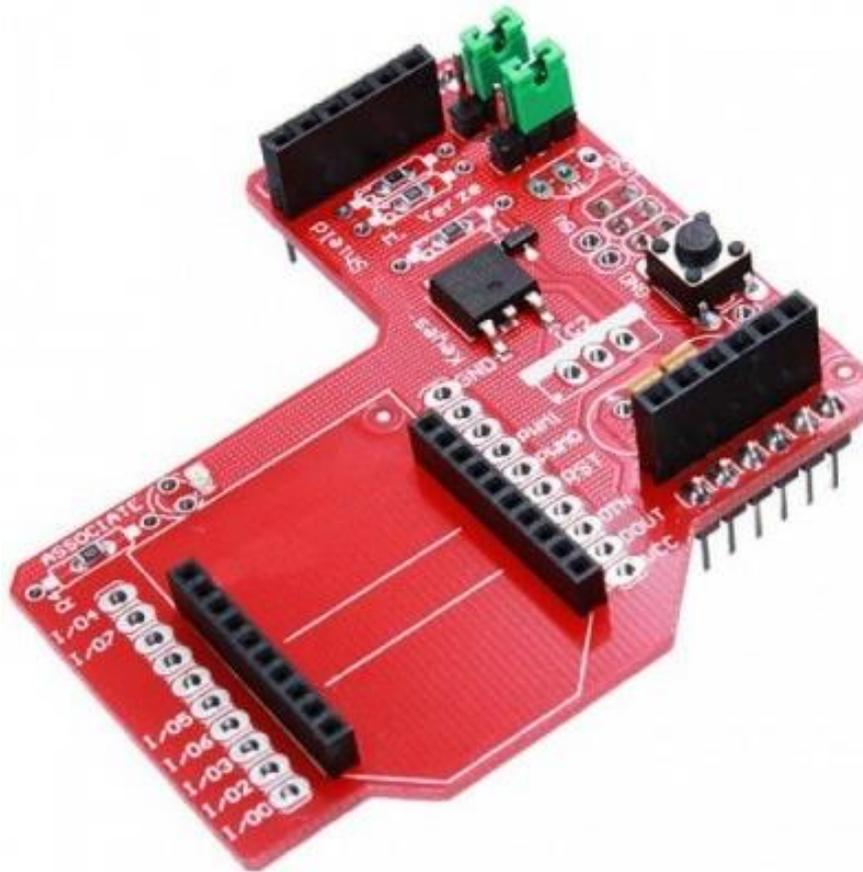


Figura 2.18 Modulo Shield Expansi3n XBee

2.12 XCTU

Para configurar y usar los m3dulos XBee es necesario descargar e instalar XBee Configuration and Test Utility (XCTU) que es un software multiplataforma que permite interactuar con los m3dulos mediante un interfaz gr3fico. Esta aplicaci3n incluye herramientas que hacen muy sencillo configurar y probar los m3dulos XBee.

XCTU nos sirve para configurar, inicializar, actualizar firmware y testear los m3dulos XBee, comunic3ndose por puerto serie a los m3dulos. Una ventaja de este software es que puedes ver r3pidamente un resumen de todos los par3metros del m3dulo y una descripci3n de ellos. (figura 2.19) [17].

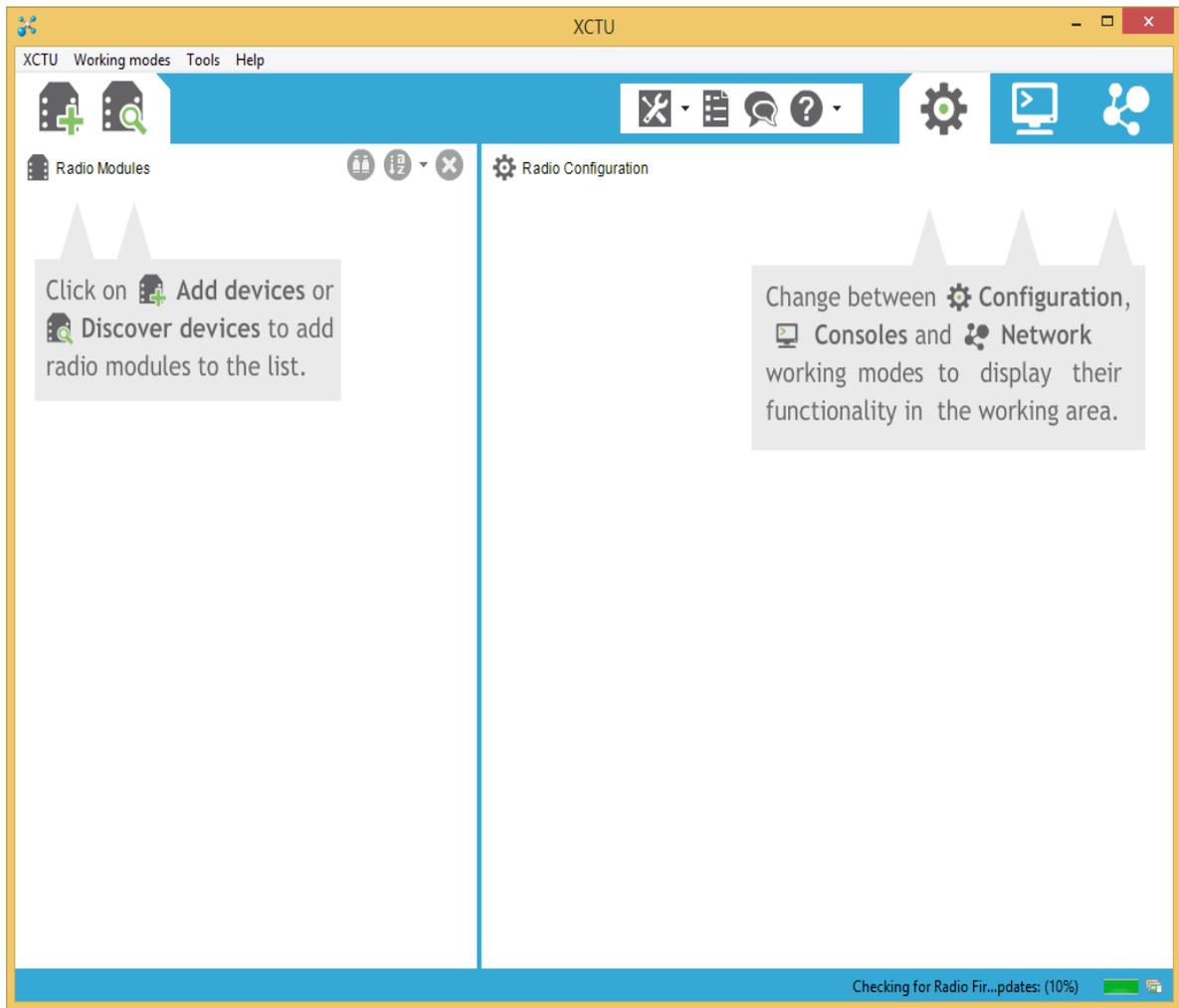


Figura 2.19 Interfaz XCTU

2.13 IDE Arduino

El entorno de desarrollo integrado también llamado IDE (sigla en inglés de Integrated Development Environment), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios lenguajes.

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación; es decir, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Además, en el caso de Arduino incorpora las herramientas para cargar el programa ya compilado en la memoria flash del hardware.

CAPÍTULO II FUNDAMENTOS TEÓRICOS

El IDE de Arduino va a ser la herramienta de trabajo para el desarrollo del prototipo e integración de Arduino con xbee y habrá que conocer su funcionamiento.

En la figura 2.20 se puede observar la estructura básica del lenguaje de programación de Arduino que se compone de al menos dos partes. Estas dos partes necesarias, o funciones, encierran bloques que contienen declaraciones, estamentos o instrucciones.

En donde `setup()` es la parte encargada de recoger la configuración y `loop()` es la que contiene el programa que se ejecutará cíclicamente (de ahí el término loop – bucle-). Ambas funciones son necesarias para que el programa trabaje.

La función de configuración (`setup`) debe contener la declaración de las variables. Es la primera función que ejecutar en el programa, se ejecuta sólo una vez, y se utiliza para configurar o inicializar `pinMode` (modo de trabajo de las E/S), configuración de la comunicación en serie y otras.

La función bucle (`loop`) siguiente contiene el código que se ejecutara continuamente (lectura de entradas, activación de salidas, etc.) Esta función es el núcleo de todos los programas de Arduino y la que realiza la mayor parte del trabajo.

La función `setup()` se invoca una sola vez cuando el programa empieza. Se utiliza para inicializar los modos de trabajo de los pines, o el puerto serie. Debe ser incluido en un programa, aunque no haya declaración que ejecutar. Así mismo se puede utilizar para establecer el estado inicial de las salidas de la placa [18].

CAPÍTULO II FUNDAMENTOS TEÓRICOS

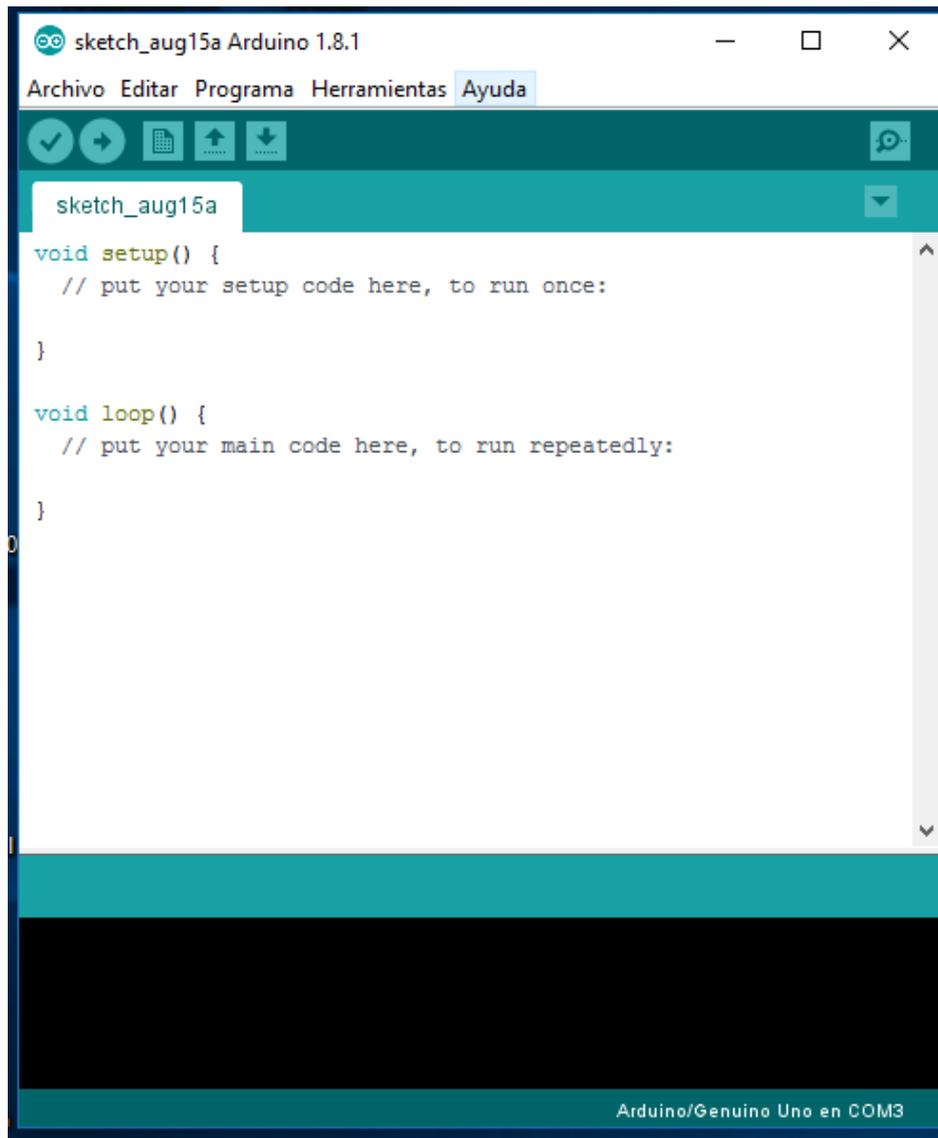


Figura 2.20 Interfaz Gráfica IDE

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

3. Desarrollo del sistema de monitoreo inalámbrico

3.1 Introducción

En este capítulo se describen los pasos realizados sobre el desarrollo del sistema de monitoreo inalámbrico

3.2 Preparación de módulos Xbee y configuración

Sobre los xbee seleccionados que en este proyecto fueron 2 XBee 2mW Wire Antenna - Series 2 (Mesh), se tiene que identificar por la parte de atrás 2 números de series y letra los cuales se anotan para posteriormente ser utilizados.

A continuación, se describen los siguientes pasos para la configuración correspondiente:

1. En la figura 3.1 se muestran los módulos correspondientes y las series a identificar.



Figura 3.1 Identificación de Series Numéricas Xbee.

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

2. Una vez identificados se procede a instalar XCTU en una Computadora para proceder a configurar cada uno de los módulos.
3. Se conecta el Explorer USB con cualquiera de los módulos Xbee y así mismo el módulo con la computadora a la que se instaló el XCTU, en la figura 3.2 se muestra un diagrama de bloques el cual representa lo anteriormente mencionado.

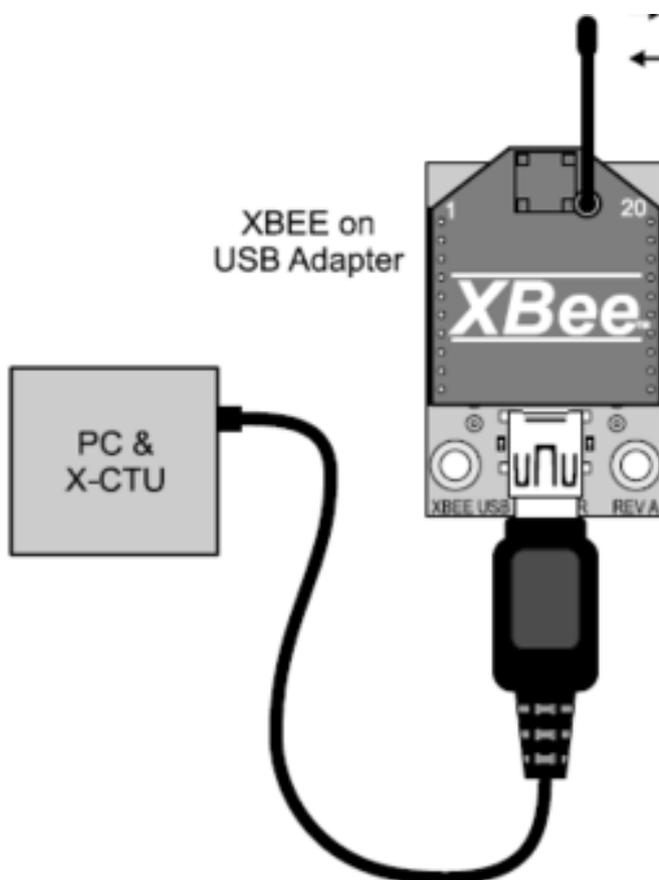


Figura 3.2 Diagrama de conexión Xbee Explorer USB, PC.

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

4. Se inicia el XCTU, para seleccionar en la lista de los puertos COM en donde se encuentre conectado nuestro dispositivo en la figura 3.3 se muestra cómo se selecciona el puerto y se pulsa el botón next para continuar con la siguiente parte.

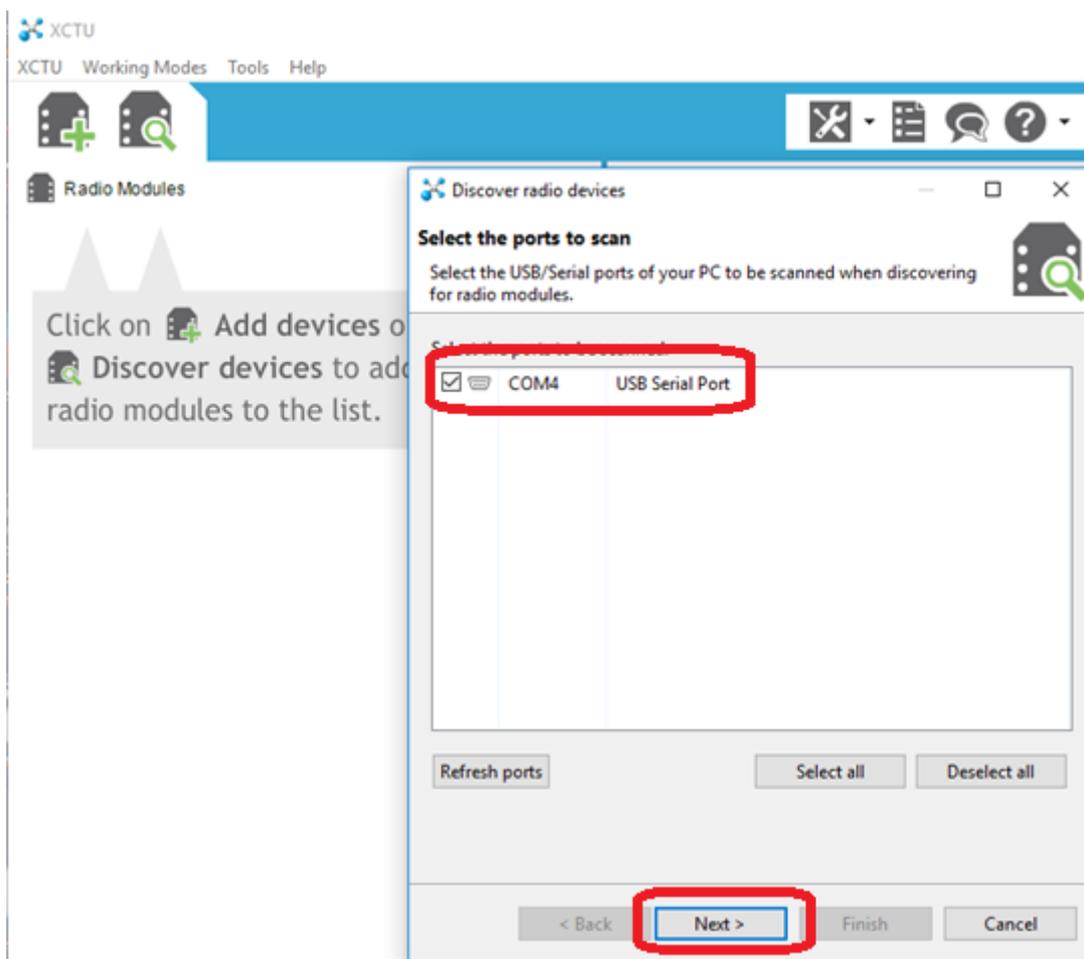


Figura 3.3 Selección de puerto COM.

5. Seleccionamos los siguientes parámetros de configuración sobre el puerto de lectura sobre el cual se va a trabajar, en la figura 3.4 se muestran los parámetros anteriormente mencionado y por último se presiona el botón Finish.

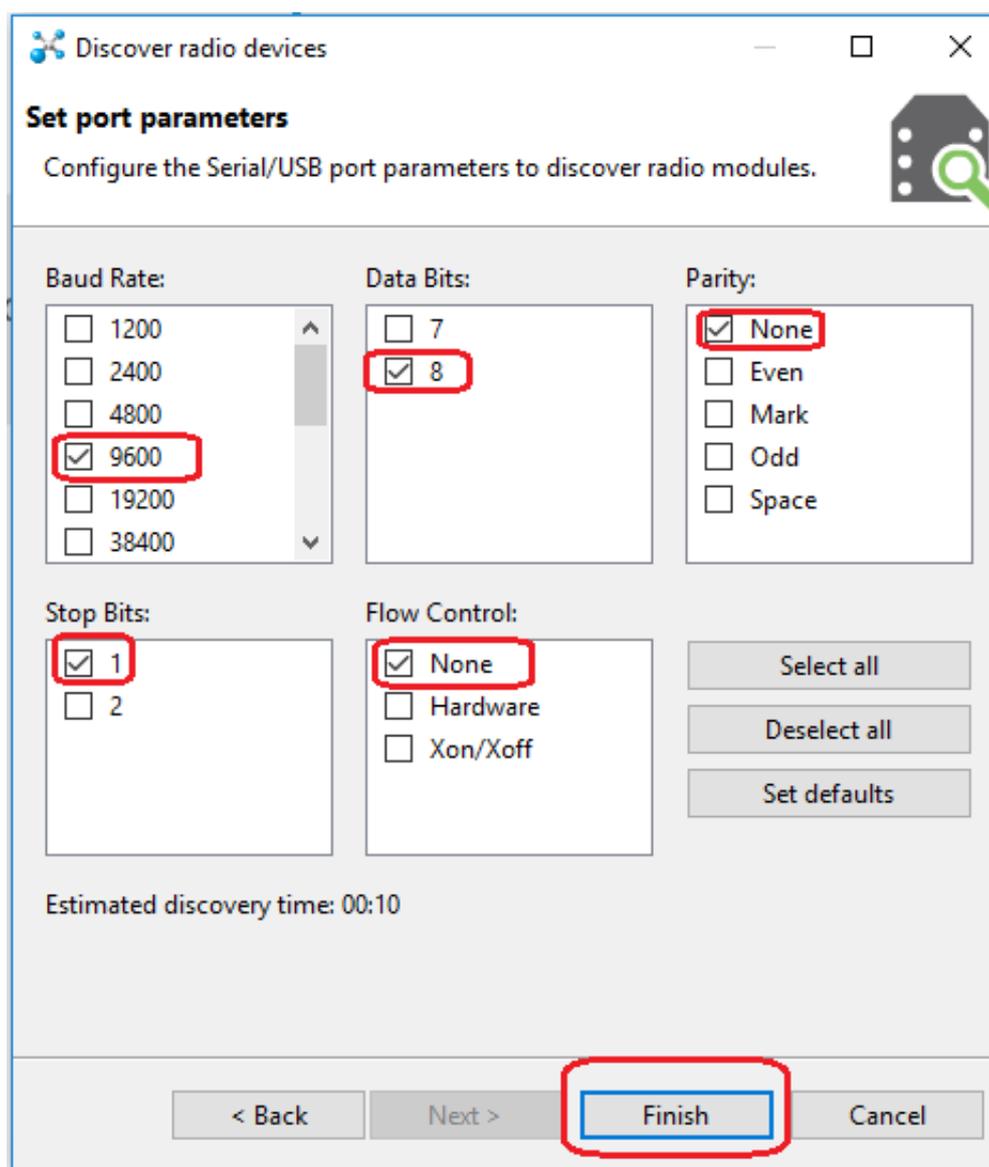


Figura 3.4 Configuración de parámetros de puerto XCTU.

6. Seleccionamos el dispositivo configurado con los parametros del paso 5 y precionamos el boton Add selected devices, en la figura 3.5 se muestra el dispositivo seleccionado es importante asegurarse de que se encuentre seleccionado antes de precionar el boton.

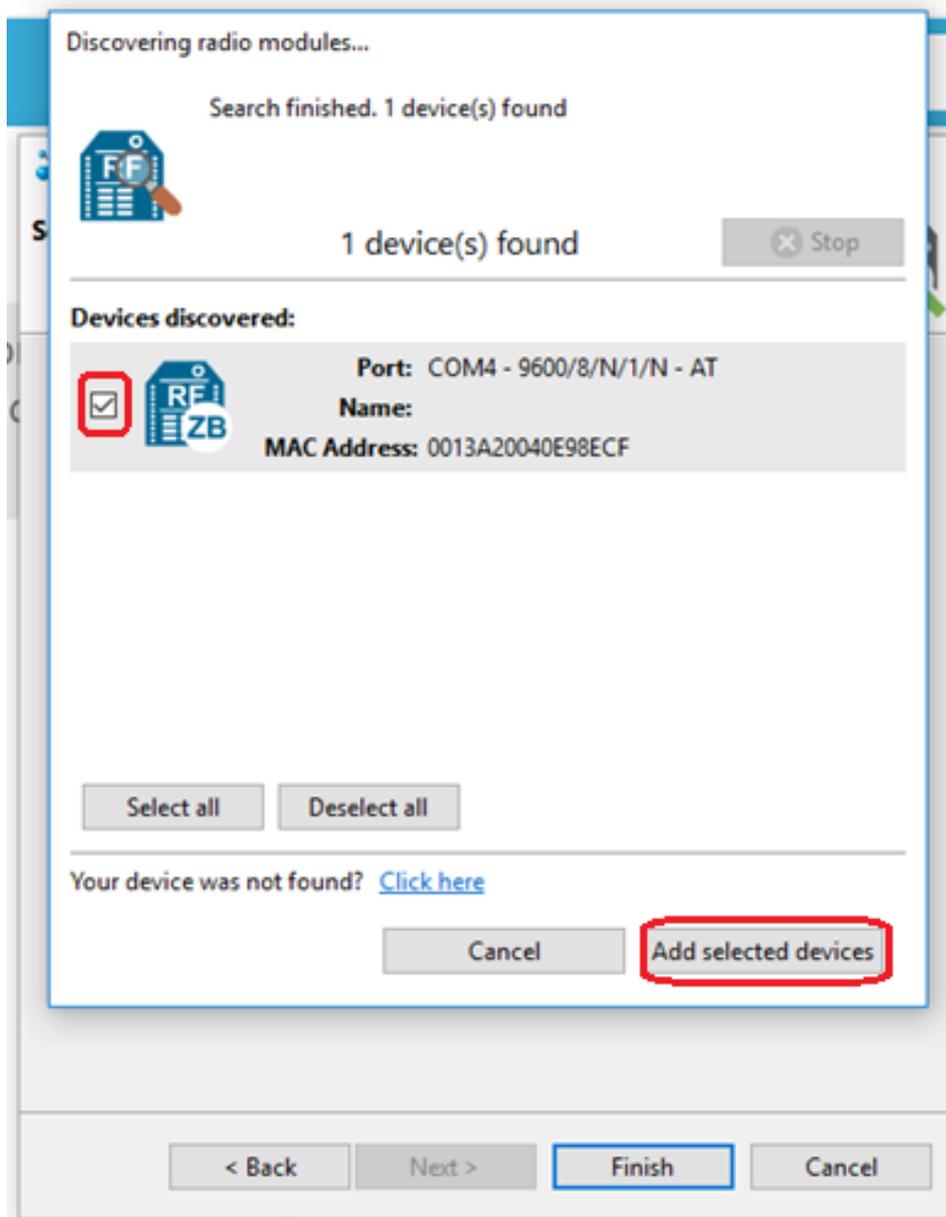


Figura 3.5 Agregar Dispositivo seleccionado.

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

7. En la figura 3.6 se ve como el dispositivo ya es agregado a la lista de módulos listo para cargar el firmware adecuado para el modelo.

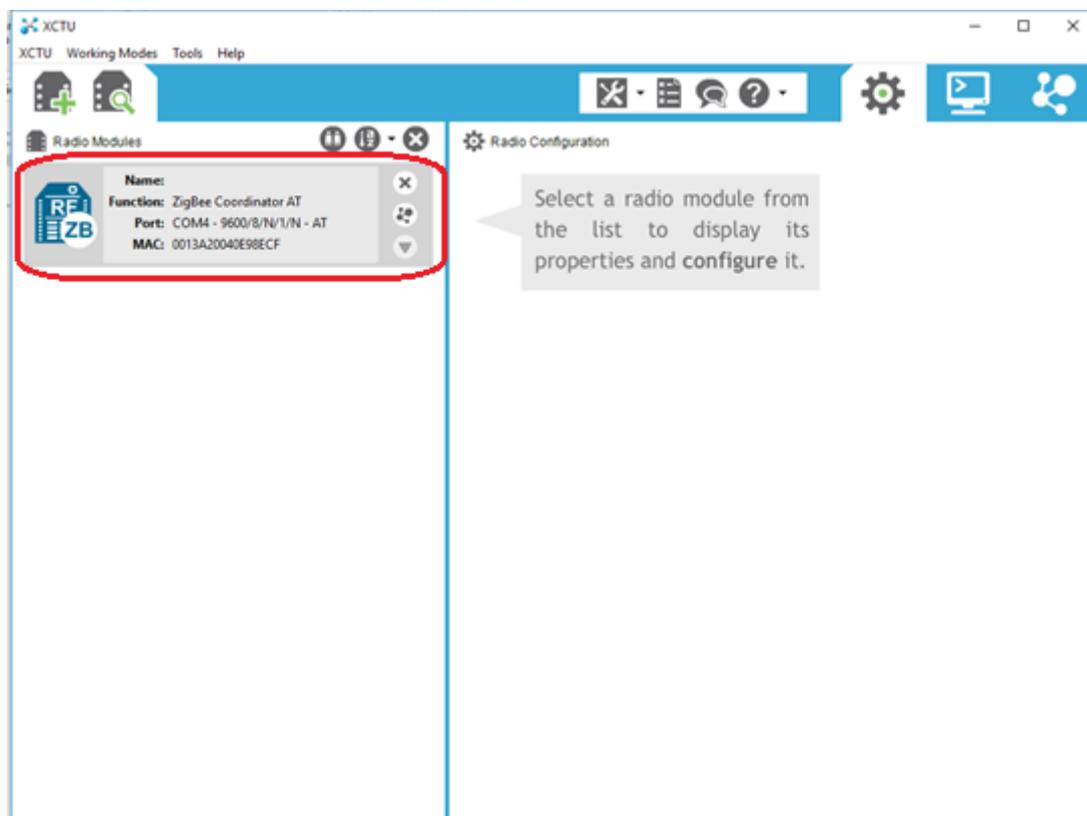


Figura 3.6 Modulo agregado.

8. Ir a la pestaña "Con el icono de herramientas", presionar "Read", elegir en "tipo" XB24-B. (en la primera configuración deberás descargar las últimas versiones), luego de eso seleccionar "ZIGBEE COORDINATOR AT" y la versión más alta que encuentres (1020), en las configuraciones siempre debe haber un coordinador, en este caso será el Xbee que se está configurando en primera instancia.

En la figura 3.7 se muestra lo anteriormente descrito sobre el punto 8.

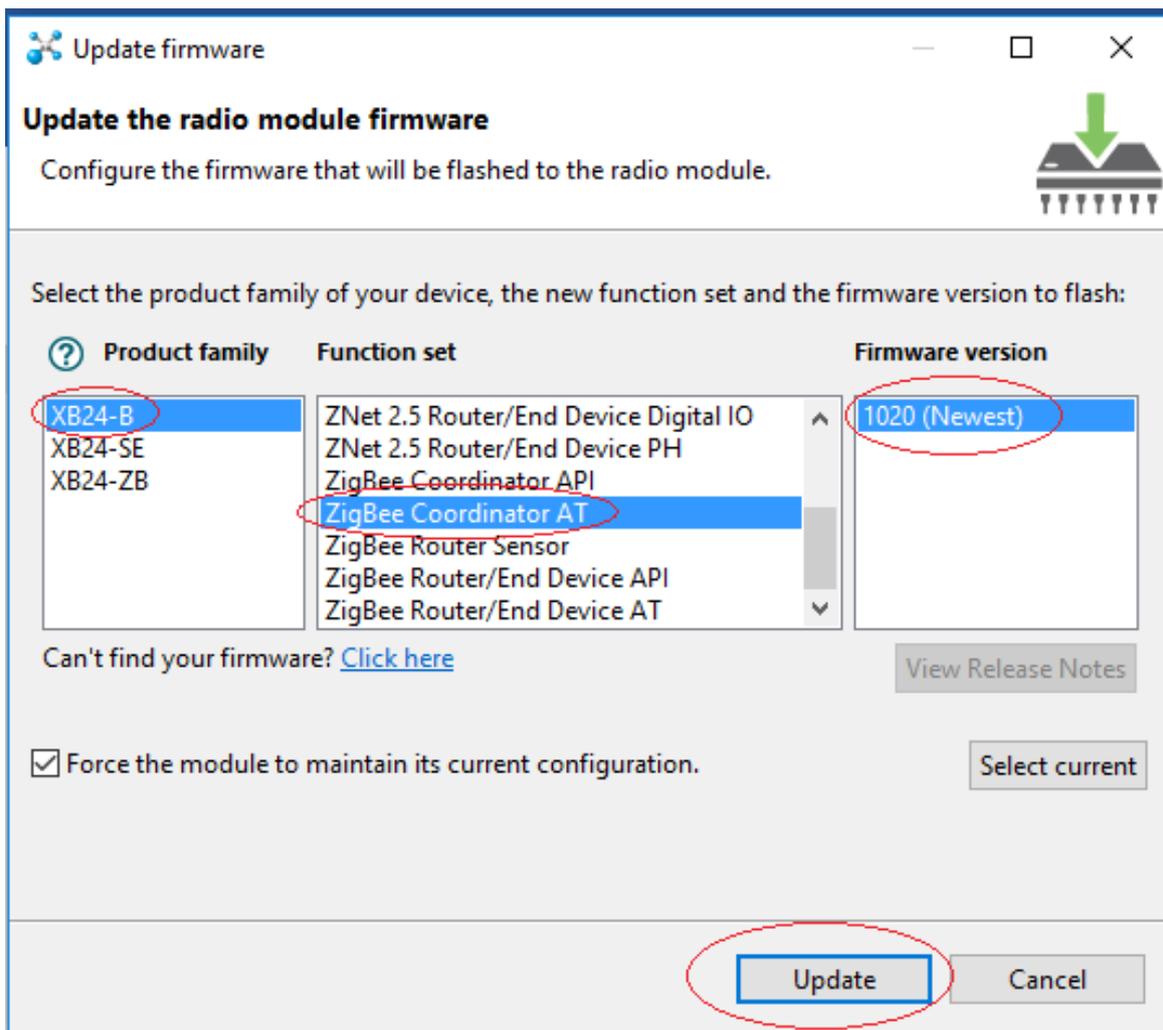


Figura 3.7 selección de Firmware Coordinador.

9. Después se identifican y cambian los siguientes valores de Networking:

- PAN ID (un número cualquiera que se le asigna a la red debe ser el mismo para los Xbee´s involucrados), en este caso seleccionamos el 205.
- Destination Address High (DH), debe ser la dirección Alta (SH) del otro XBee (0013A200 en este caso, el número de arriba en Xbee) observar figura 3.1.

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

- Destination Address Low (DL), debe ser la dirección Baja (SL) del otro XBee. (el número de abajo) observar figura 3.1.
- Finalmente actualizar los valores seleccionados en el botón "Write", y ya tenemos el módulo coordinador configurado el cual se da la tarea de un receptor de datos.

En la Figura 3.8 podemos observar los valores que se configuraron.

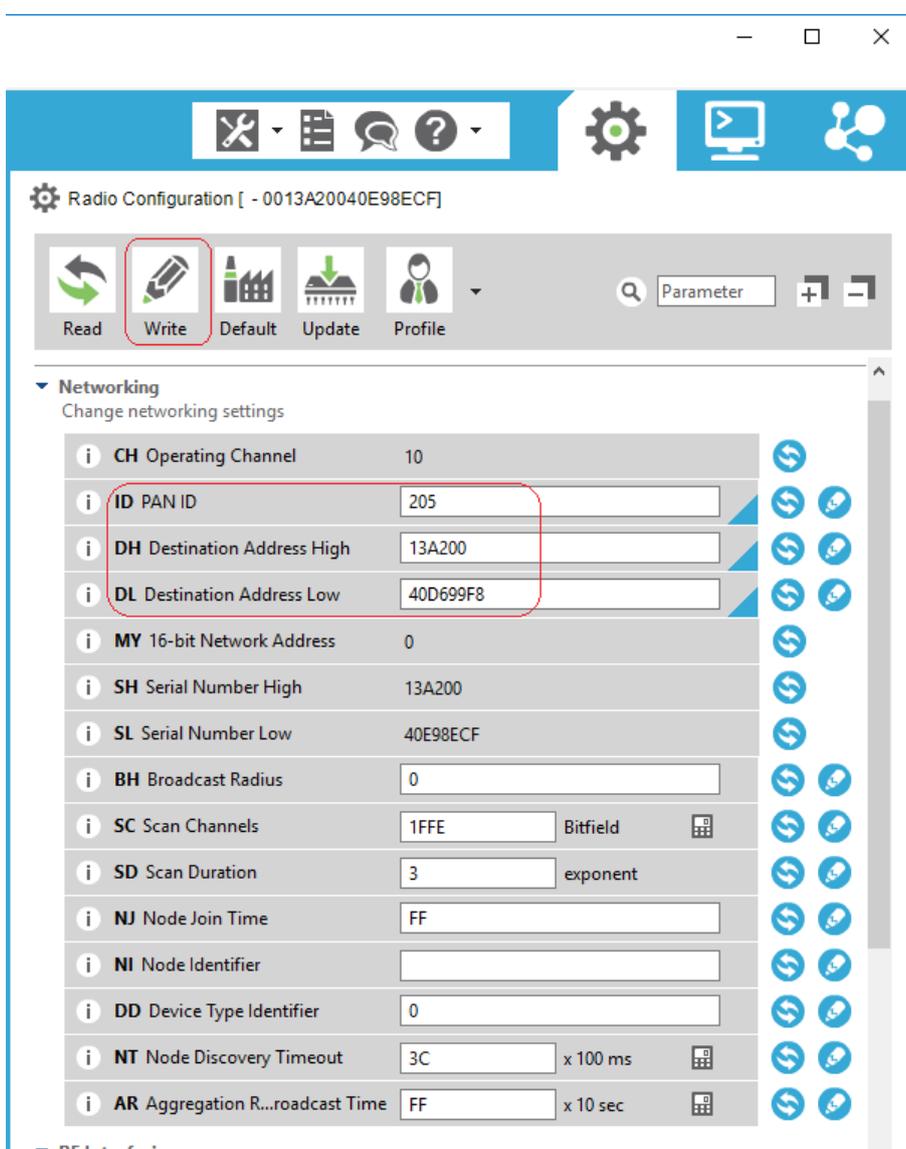


Figura 3.8 Configuración de Valores Networking Coordinador.

10. Conectamos el otro Xbee en el Explorer Regulated USB para configurarlo como nuestro emisor de datos por lo cual hacemos lo siguiente:

- Para hacer la configuración del módulo restante se repiten los pasos 3, 4, 5, 6, 7 y en el paso 8 en vez de elegir "Coordinator" elegimos "ZNet 2.5 ROUTER/END DIVICE AT". Como se muestra en la figura 3.9

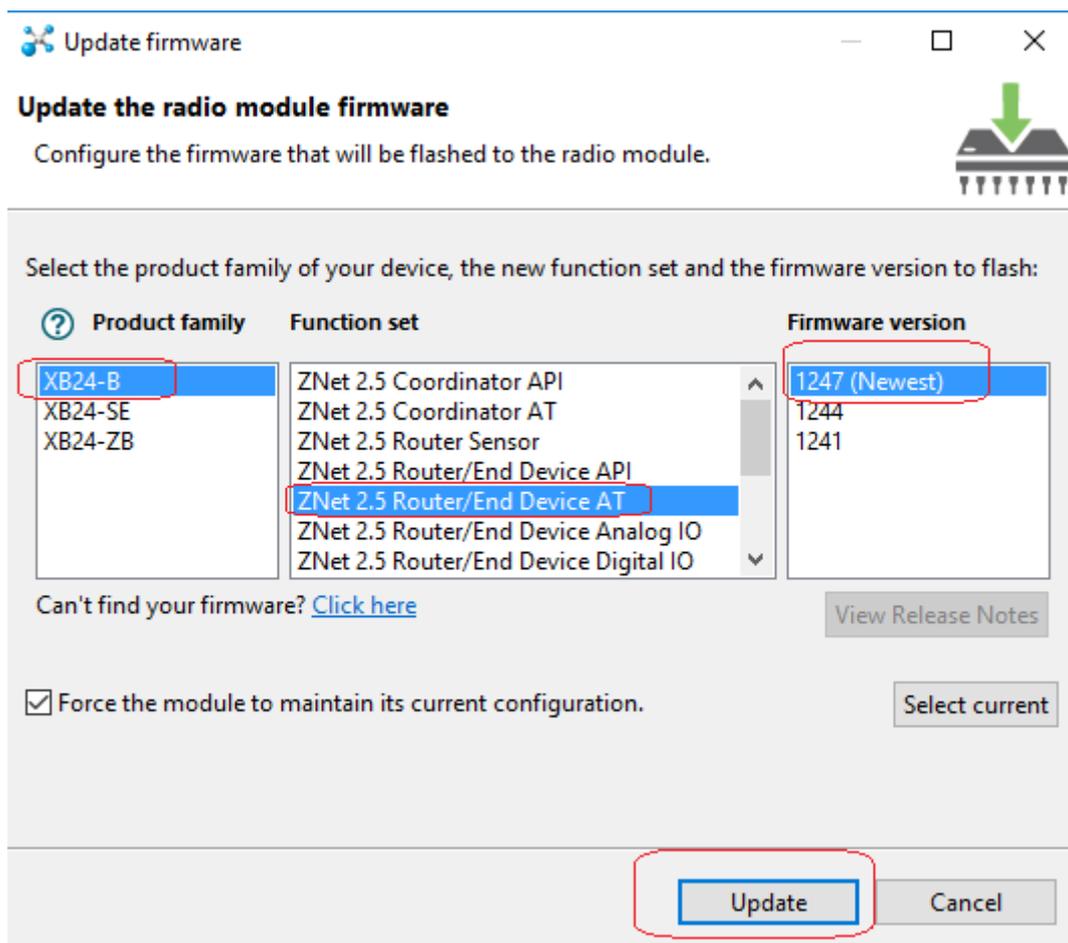


Figura 3.9 Configuración Firmware Router.

- En el paso 9 se selecciona el mismo valor asignado para PAN ID (205), además los valores DH y DL ahora son respecto al Xbee anterior. (figura 3.10).

Por último, identificamos el valor JV al cual se asigna 1 (Enabled) y presionamos "Write", como se muestra en la figura 3.10 [19].

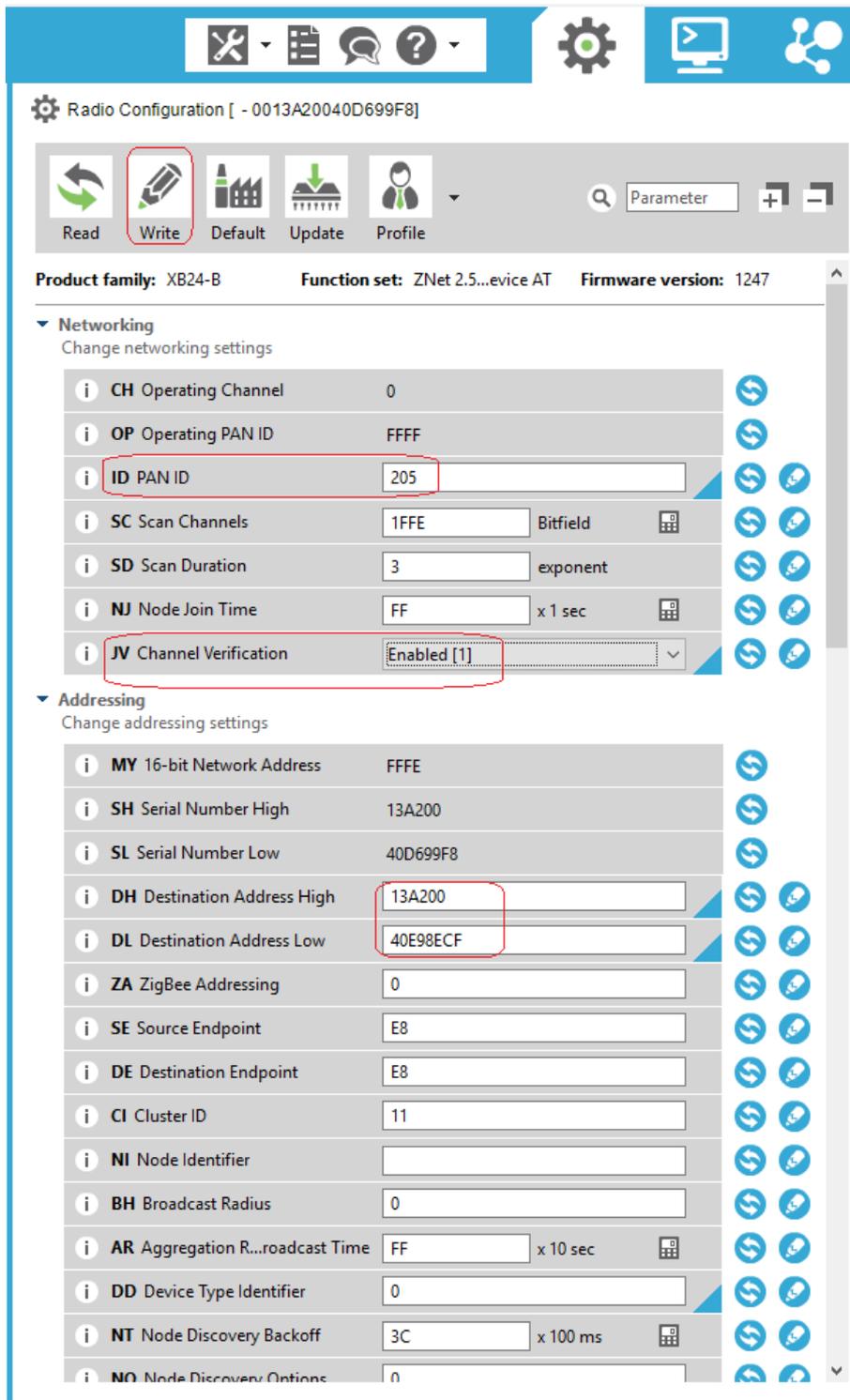


Figura 3.10 Configuración de Valores Networking Router.

3.3 Conexión de Arduino UNO con Xbee emisor

Para la conexión y configuración del módulo emisor se tiene que realizar una conexión entre el módulo xbee y el Arduino UNO, para ello se utiliza el Módulo Shield Expansión XBee el cual quedara montado sobre los pines A0, A1, A2, A3, A5, TX, RX, 1, 2, 3, 4, 5, 6, 7, y los 6 pines del ISCP, de todos los mencionados solo TX y RX quedan tapados los demás se elevan a la superficie para poder ser utilizados con normalidad, en la figura 3.11.

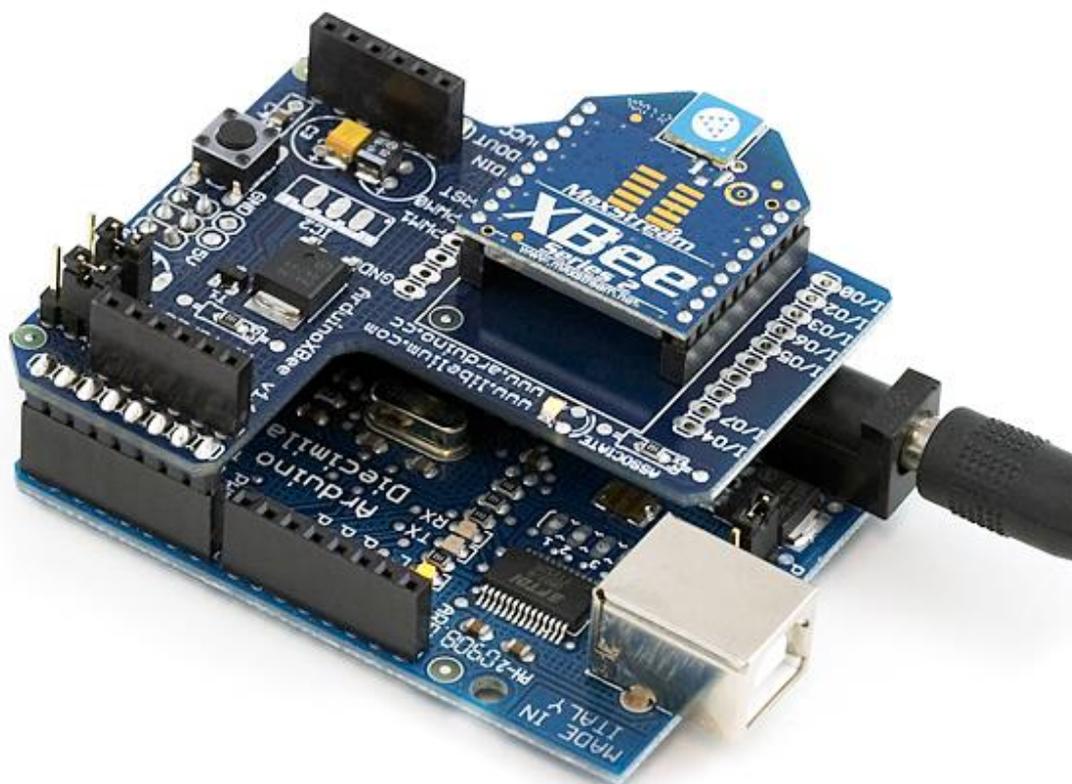


Figura 3.11 Arduino y Shield Expansión XBee

Antes de cargar cualquier programación al Arduino, es muy importante quitar el Xbee, ya que causa errores a la hora de cargar los programas, el shield puede quedarse conectado no causa falla alguna ya que prácticamente todos los pines digitales y analógicos quedan disponibles para los sensores [20].

3.4 Conexión sobre red de sensores DS18B20

Para esta conexión se determinó que uno de los sensores mejor adaptados para el propósito del proyecto sería el DS18B20, para ellos se hicieron pruebas simulando la red con 2 sensores de este tipo conectados en serie ya que el funcionamiento de lectura de este tipo de sensores tiene 2 formas diferentes para realizar dicha lectura, Tenemos dos opciones o métodos que podemos usar cuando necesitamos leer más de un sensor de temperatura.

El primer método es manejando cada sensor con un pin diferente del Arduino. De esta forma sí tenemos 3 sensores, necesitaremos usar 3 pines digitales del Arduino, este método quita espacio en el Arduino ya que solo cuenta con pocos pines digitales en caso de conectar más de 6 sensores no funcionaría así que esta opción no se tomó viable para hacer la toma de lectura de los sensores.

Otra forma es usar el mismo pin para todos los sensores, en otras palabras, todos los sensores se conectan al mismo bus 1-Wire, y como cualquier bus, cada elemento o dispositivo posee una identificación o dirección. En el caso del DS18B20 necesitamos averiguar su dirección que es única y viene establecida de fábrica.

Para ellos se utiliza este pequeño programa realizado en el IDE de Arduino:

```
#include <OneWire.h> // librería que predefinida para el sensor utilizado
```

```
OneWire ourWire(2); //Se establece el pin 2 como bus OneWire
```

```
void setup(void) {
```

```
  Serial.begin(9600); //inicialización del puerto de comunicación
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
}
```

```
void loop(void) {
```

```
    byte addr[8];          //variable definida para almacenar el dato solicitado
```

```
    Serial.println("Obteniendo direcciones:"); //impresión de caracteres en puerto serial
```

```
    while (ourWire.search(addr)) //búsqueda y almacenamiento sobre la variable definida
```

```
    {
```

```
        Serial.print("Address = "); // impresión de carácter
```

```
        for( int i = 0; i < 8; i++) { //inicialización de ciclo para preguntar la dirección sobre cada sensor conectado
```

```
            Serial.print(" 0x");
```

```
            Serial.print(addr[i], HEX);          //almacenar el resultado hexadecimal en la variable definica
```

```
        }
```

```
        Serial.println(); //imprimir con salto de línea
```

```
    }
```

```
Serial.println();
```

```
ourWire.reset_search();
```

```
delay(2000); //esperar 2000mil milisegundos para volver a preguntar a todos de nuevo
```

```
}
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

Después de instalar e iniciar el IDE de Arduino se tienen que cargar las librerías correspondientes anteriormente mencionadas, una vez echo copia y pega el programa en el IDE de Arduino y se verifica que no tenga ningún error de sintaxis en el botón verificar como se muestra en la figura 3.12.



```
direccionesbuscador Arduino 1.8.5
Archivo Editar Programa Herramientas Ayuda
direccionesbuscador
#include <OneWire.h>

OneWire ourWire(2); //Se establece el pin 2 como bus OneWire

void setup(void) {
  Serial.begin(9600);
}

void loop(void) {
  byte addr[8];
  Serial.println("Obteniendo direcciones:");
  while (ourWire.search(addr))
  {
    Serial.print("Address = ");
    for( int i = 0; i < 8; i++) {
      Serial.print(" 0x");
      Serial.print(addr[i], HEX);
    }
    Serial.println();
  }

  Serial.println();
  ourWire.reset_search();
  delay(2000);
}

Compilado
El Sketch usa 2672 bytes (8%) del espacio de almacenamiento de programa. El máximo
Las variables Globales usan 240 bytes (11%) de la memoria dinámica, dejando 1808
```

Figura 3.12 Verificación del Programa de Direcciones.

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

Una vez cargado el programa se hace la conexión del Arduino con los sensores que deseamos identificar, como se puede ver en la figura 3.13 se conectan las salidas de los sensores a el pin digital “2” conectados con una resistencia con un valor de $4.7k\Omega$

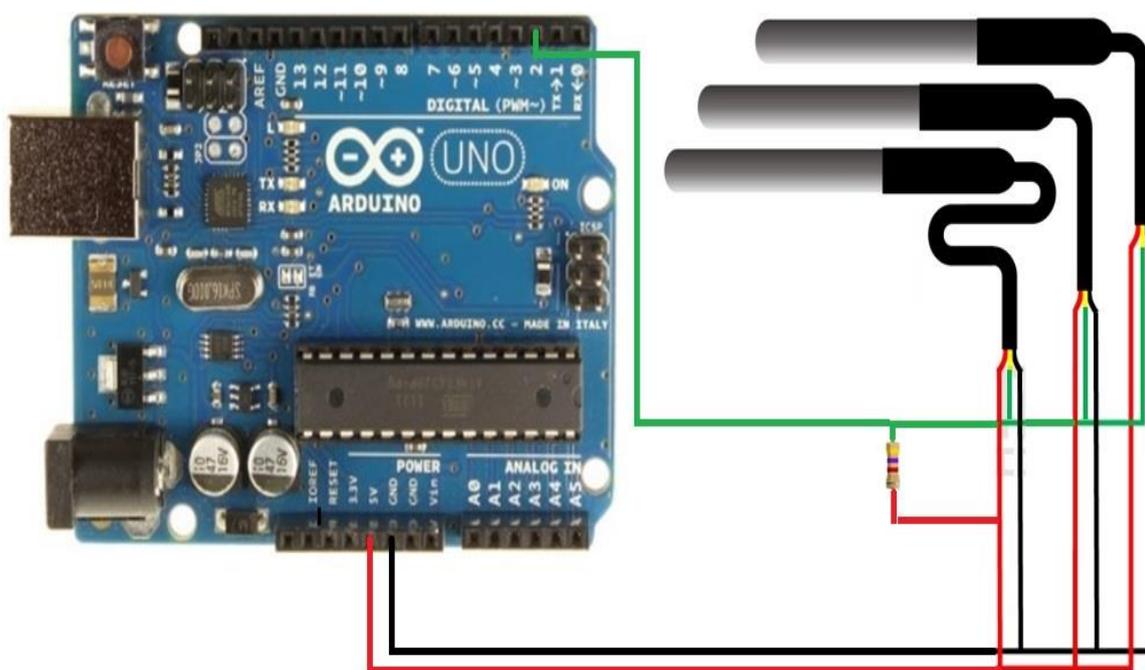


Figura 3.13 Diagrama de conexión Sensores y Arduino

Conectar el Arduino con el cable USB/serial al puerto COM que se desee, en la parte superior en el menú de herramientas se carga el tipo de tarjeta con la que se está trabajando, en este caso es Arduino UNO y el puerto por el cual se está comunicando es el “COM4”, en la figura 3.14 se observa la selección de los parámetros anterior mente descritos.

Una vez que se seleccionaron las placa y el puerto de comunicación correctos se carga el programa en el microcontrolador, si no existe ningún error la ventana que aparece es igual a la de la figura 3.12, para cargar el programa en la tarjeta solo tenemos que presionar el botón con una flecha apuntando hacia la derecha.

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

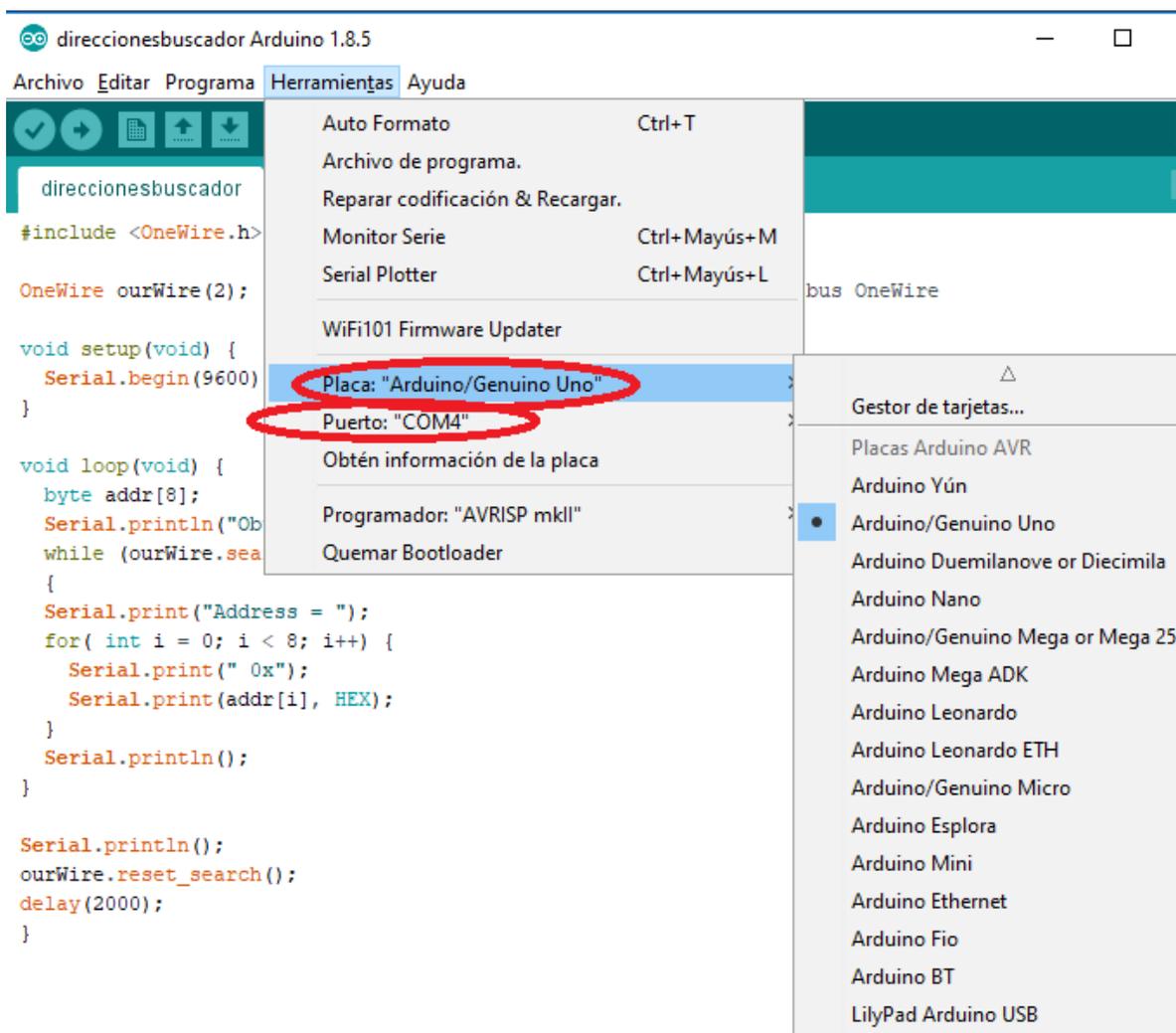


Figura 2.14 Selección de Placa y puerto en IDE

Presionamos el visor de puerto serial en la parte superior derecha, para poder visualizar los datos que necesitamos sobre las direcciones de identificación de los sensores conectados en la siguiente figura 2.15 se observa lo mencionado cabe mencionar que esta prueba se hizo con 2 sensores por lo tanto solo aparecen 2 direcciones.

El código es el mismo para 2 para 3 para 8 ya que el ciclo del programa está diseñado para que haga las peticiones necesarias hasta terminar con las 8 peticiones, si se conectan más es necesario modificar el 8 en el ciclo del programa por el número deseado [13].

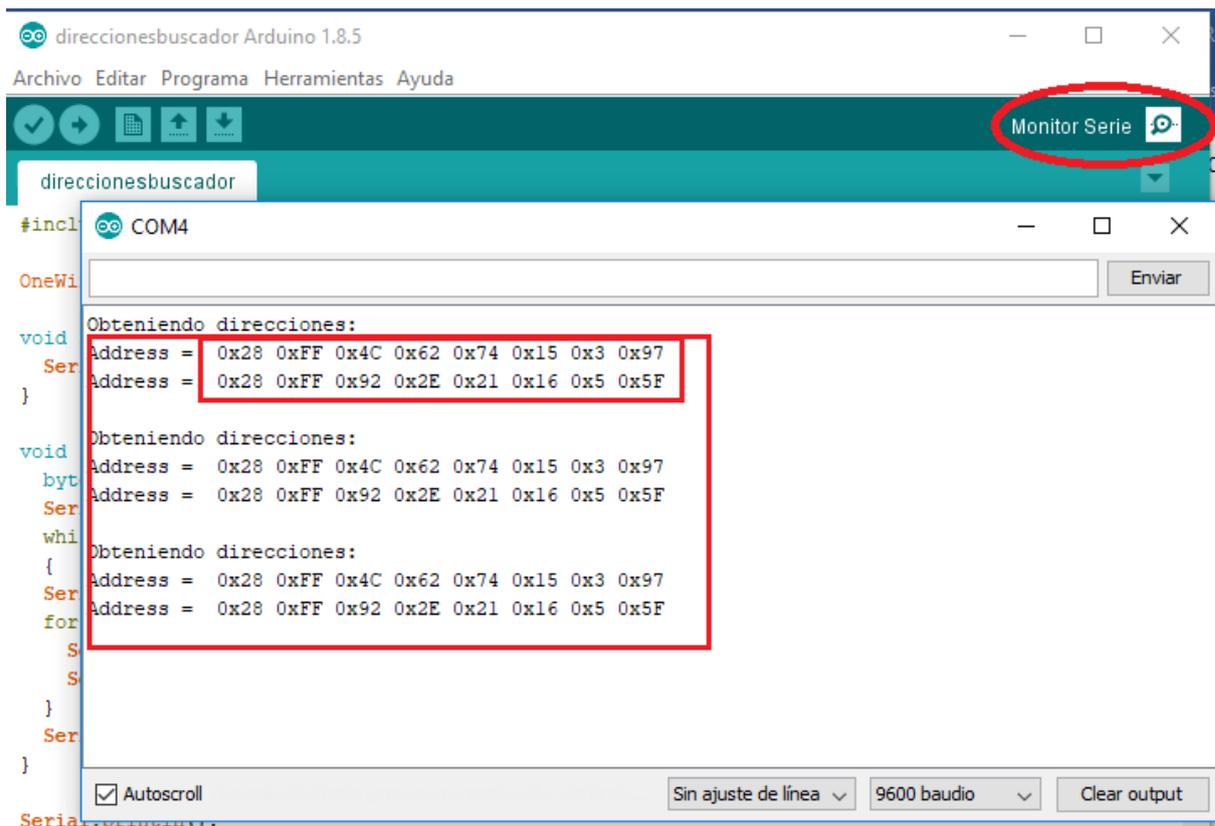


Figura 3.15 Visor de puerto serial.

3.5 Programación del módulo emisor Arduino Uno, Xbee y Sensor DS18B20

Las direcciones en hexadecimal son las que se van a utilizar para pedir la lectura ya que lo anterior solo se realizó para obtener las direcciones sobre las cuales se le pedirá censar a cada uno de los sensores.

Por último, se diseñó el siguiente código para enviar la lectura de los sensores:

```
// librerias sensor Dsb20

#include <OneWire.h>

#include <DallasTemperature.h>

// Sensor de Temperatura
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
OneWire ourWire(2); //Se establece el pin 2 como bus OneWire
```

```
DallasTemperature sensors(&ourWire); //Se declara una variable sobre las  
direcciones obtenidas anteriormente
```

```
DeviceAddress address1 = {0x28, 0xFF, 0x4C, 0x62, 0x74, 0x15, 0x3,  
0x97}; //dirección del sensor 1
```

```
DeviceAddress address2 = {0x28, 0xFF, 0x92, 0x2E, 0x21, 0x16, 0x5,  
0x5F}; //dirección del sensor 2
```

```
//declaración de variables para determinar la temperatura
```

```
int conta;
```

```
int tprom;
```

```
//*****
```

```
void setup() {
```

```
  //delay(10);
```

```
  Serial.begin(9600);
```

```
  sensors.begin(); //Se inicia el sensor
```

```
}
```

```
void loop() {
```

```
  /// adquisición de datos
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
conta=0;//se inicializa la variable en 0
```

```
    sensors.requestTemperatures();    //envía el comando para obtener las
temperaturas
```

```
int temp1= sensors.getTempC(address1);//Se obtiene la temperatura en °C del
sensor 1
```

```
if(temp1>=0){
```

```
    conta=conta+1;//
```

```
}
```

```
    else{temp1=0;} //cunado uno de los sensores tiene un falso o no se encuentra
conectado por lo regular marca un valor negativo
```

```
int temp2= sensors.getTempC(address2);//Se obtiene la temperatura en °C del
sensor 2
```

```
if(temp2>=0){
```

```
    conta=conta+1;
```

```
}
```

```
    else{temp2=0;}
```

```
if(conta>0){
```

```
    //Ejemplo si se tuviera una red de sensores de 16
```

```
    //tprom=(temp1+temp2+temp3+temp4+temp5+temp6+temp7+temp8+temp9+temp
10+temp11+temp12+temp13+temp14+temp15+temp16)/conta;
```

```
    tprom=(temp1+temp2)/conta;
```

```
}
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
else {  
    Serial.println(tprom=0);  
}  
Serial.println(tprom);  
  
String tp= String(tprom,DEC);  
delay(1000);  
}
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

El programa anterior, tiene como objetivo el tomar la lectura de cada sensor y promediarla para obtener la temperatura ambiente del lugar, así como para poder enviarla por el dispositivo Xbee, y representarla en una pantalla TFT.

Para cargar el programa es exactamente igual que como se realizó la carga del programa para obtener las direcciones de los sensores, la conexión con el Arduino es igual solo para que funcione con el Xbee se tiene que poner antes el shield expansor junto con el Xbee en el área correspondiente.

En la figura 3.16 se puede observar la conexión final del dispositivo que será el emisor de los datos de la red de sensores, este dispositivo sería colocado dentro del invernadero [18].

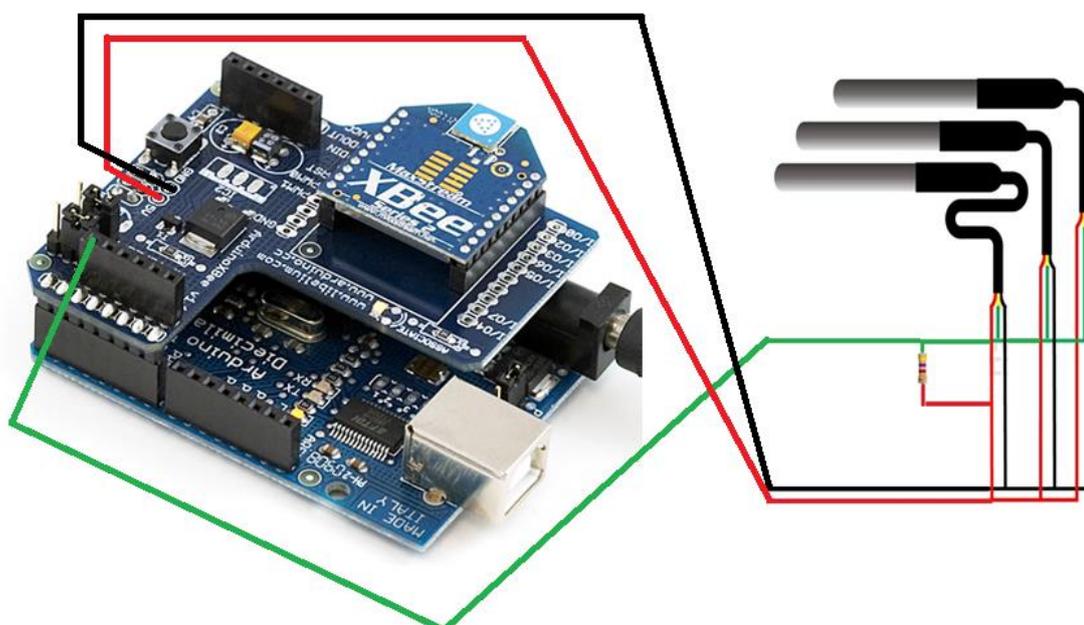


Figura 3.16 Representación de Conexión Final Emisor

3.6 Conexión de Arduino MEGA con pantalla TFT

Par la conexión del Arduino mega y la pantalla TFT se necesita únicamente colocar la pantalla en los pines correctos de la tarjeta microcontroladora ya que la estructura de la pantalla ya está hecha tanto como para Arduino MEGA como para Arduino uno los pines que se ocupan de la tarjeta micro controladora son los siguientes:

GND, pines Digitales 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, TX, RX, pines analógicos A5, A4, A3, A2, A1, A0, Vin, GND, GND, 5v, 3.3v, en la figura 3.17 se puede observar gráficamente los pines ocupados por la pantalla TFT en Arduino Mega.

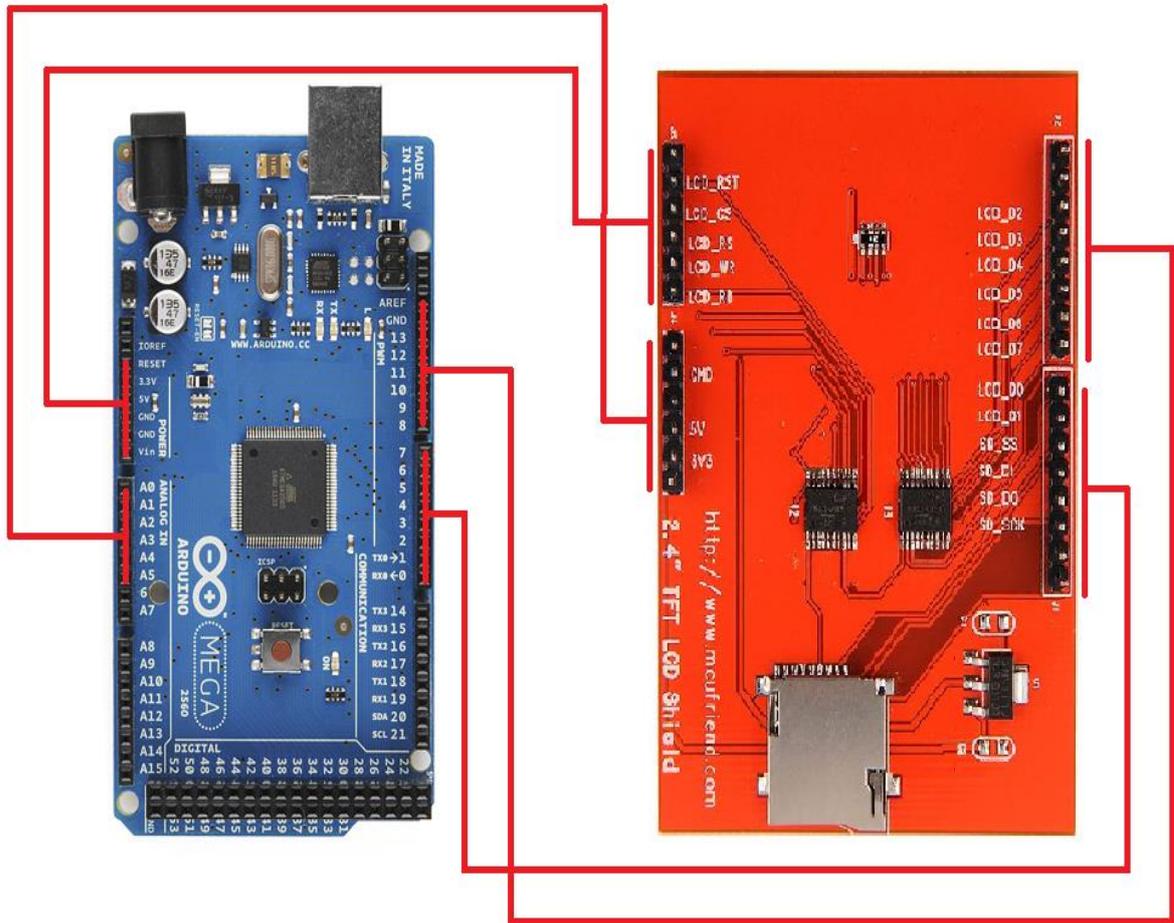


Figura 3.17 Conexión Arduino Mega Pantalla TFT.

3.7 Conexión entre Arduino Mega y Xbee

Para la conexión es necesario usar el mismo xbee Explorer solo que en vez de utilizar el puerto serial se utilizan los pines adicionales los cuales son 5v, GND, TX, RX, en la figura 3.18 se puede apreciar como los pines anteriormente mencionados

se conectan a los pines de la tarjeta microcontroladora Arduino Mega los pines q se ocupan de esta tarjeta son GND, 5V, pin Digital 50, 51.

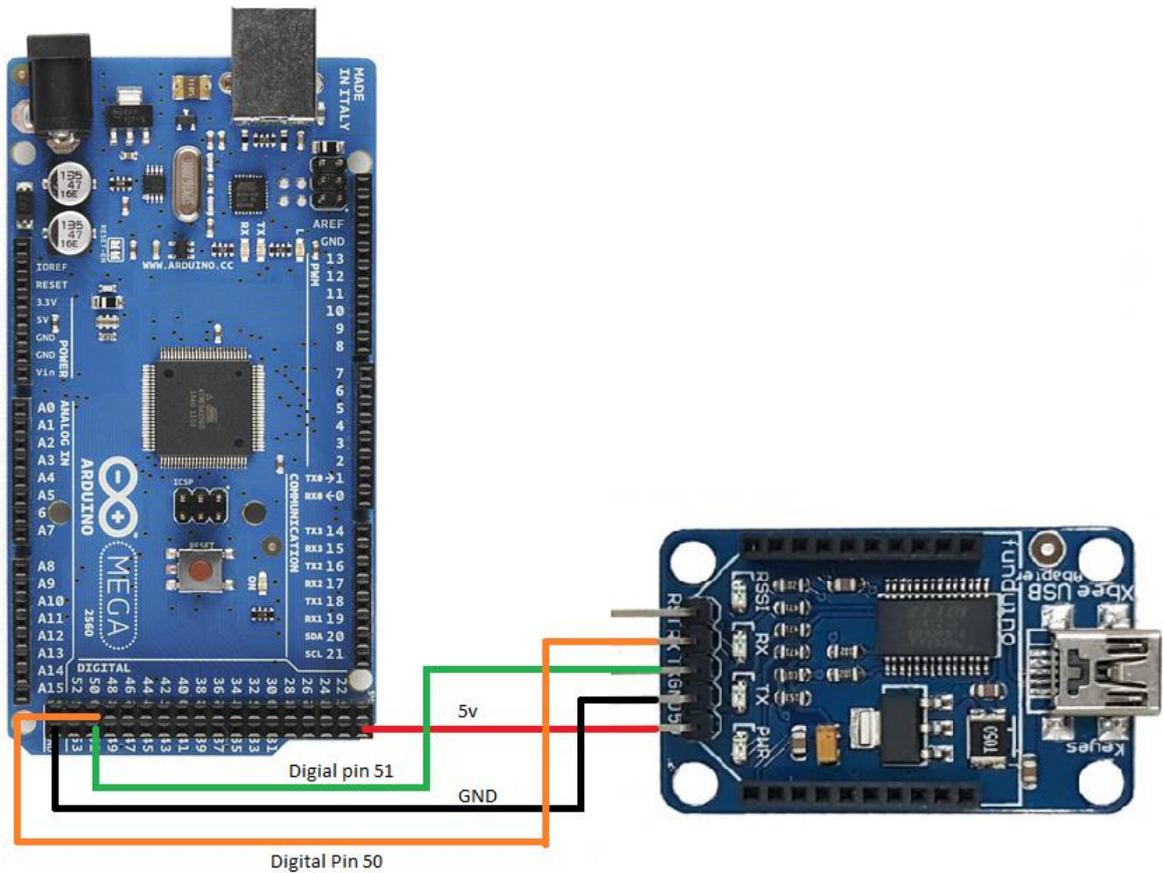


Figura 3.18

Teniendo este acomodo facilita que la pantalla TFT Funcione correctamente y no haya ningún inconveniente para que trabajen el conjunto.

3.8 Programación del módulo Receptor Arduino MEGA, Xbee y TFT

El siguiente programa se utilizaron 5 librería importantes tanto para la representación gráfica de los datos, como para la recepción de los datos de temperatura enviados desde el módulo emisor.

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
#include <Adafruit_GFX.h> // librería del control grafico

#include <Adafruit_TFTLCD.h> // Librería para el control del hardware

#include <TouchScreen.h> // librería para el control tactil

#include <Wire.h> //librería encargada de la comunicación con dispositivos i2c(xbee
//Explorer)

//#include <DallasTemperature.h> //librería encargada del sensor utilizado

#include <SoftwareSerial.h> // librería encargada de la comunicación serial

#define DEBUG(a) Serial.print(a);

SoftwareSerial mySerial(50, 51); // RX, TX pines de Arduino mega

// *** se definen los pines que controlaran el táctil de la pantalla tft

#define YP A3

#define XM A2

#define YM 9

#define XP 8

// *** Define las dimensiones del el táctil que se van a evaluar cuando reciba una
//pulsación

#define TS_MINX 150

#define TS_MINY 120

#define TS_MAXX 920

#define TS_MAXY 940
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
TouchScreen ts = TouchScreen(XP, YP, XM, YM, 300);
```

```
// *** Definición de pines que se usaran para la LCD
```

```
#define LCD_CS A3
```

```
#define LCD_CD A2
```

```
#define LCD_WR A1
```

```
#define LCD_RD A0
```

```
#define LCD_RESET A4
```

```
// *** definición de los nombres de los colores que se pueden utilizar
```

```
#define BLACK 0x0000
```

```
#define WHITE 0xFFFF
```

```
#define RED 0xF800
```

```
#define GREEN 0x07E0
```

```
#define BLUE 0x001F
```

```
#define CYAN 0x07FF
```

```
#define MAGENTA 0xF81F
```

```
#define YELLOW 0xFFE0
```

```
#define GREY 0x2108
```

```
#define MINPRESSURE 10
```

```
#define MAXPRESSURE 1000
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
Adafruit_TFTLCD tft(LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);

// Definición de esquemas para los colores

#define RED2RED 0

#define GREEN2GREEN 1

#define BLUE2BLUE 2

#define BLUE2RED 3

#define GREEN2RED 4

#define RED2GREEN 5

uint32_t runTime = -99999;    // hora de la próxima actualización

int reading = 0; // Valor que se mostrará

int d = 0; // Variable utilizada para la forma de onda de prueba sinusoidal

boolean alert = 0;

int8_t ramp = 1;

int tesmod =1;

// Cuerpo del programa, inicializando puertos de comunicación de todos los
//dispositivos que se van a utilizar

void setup() {

  mySerial.begin(9600);

  Serial.setTimeout(50);

  Wire.begin();

  tft.reset();

  Serial.begin(9600);

  digitalWrite(35, HIGH);    //Se usa para la retro iluminación del LCD
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
tft.setRotation(1);

progmemPrintln(PSTR("TFT LCD test"));

tft.fillScreen(BLACK);

#ifdef USE_ADAFRUIT_SHIELD_PINOUT

  progmemPrintln(PSTR("Using Adafruit 2.8\" TFT Arduino Shield Pinout"));
#else

  progmemPrintln(PSTR("Using Adafruit 2.8\" TFT Breakout Board Pinout"));
#endif

tft.reset();

uint16_t identifier = tft.readID();

if(identifier == 0x9325) {

  progmemPrintln(PSTR("Found ILI9325 LCD driver"));
} else if(identifier == 0x9328) {

  progmemPrintln(PSTR("Found ILI9328 LCD driver"));
} else if(identifier == 0x7575) {

  progmemPrintln(PSTR("Found HX8347G LCD driver"));
} else {

  progmemPrint(PSTR("Unknown LCD driver chip: "));

  Serial.println(identifier, HEX);

  progmemPrintln(PSTR("If using the Adafruit 2.8\" TFT Arduino shield, the line:"));

  progmemPrintln(PSTR(" #define USE_ADAFRUIT_SHIELD_PINOUT"));
}
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
progmemPrintln(PSTR("should appear in the library header (Adafruit_TFT.h)."));  
progmemPrintln(PSTR("If using the breakout board, it should NOT be  
#defined!"));  
  
progmemPrintln(PSTR("Also if using the breakout, double-check that all wiring"));  
progmemPrintln(PSTR("matches the tutorial."));  
  
return;  
}
```

```
tft.begin(identifiser);  
  
    //definición de colores utilizados en delimitaciones  
  
tft.fillScreen(BLACK); //color de fondo  
  
tft.setCursor (12,208);  
  
tft.setTextSize (2);  
  
tft.setTextColor (WHITE,BLACK);  
  
tft.print ("RUBEN MEDINA ARIAS");  
  
tft.setCursor (75,78);  
  
tft.setTextSize (1);  
  
tft.setTextColor (WHITE,BLACK);  
  
tft.print ("TEMPERATURE");  
  
}
```

```
void loop() {  
  
float t;    // variable de recepción de temperatura  
  
while(1){
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
if (mySerial.available()){
```

```
    t = mySerial.parseInt();
```

```
    DEBUG((int)t);
```

```
}
```

```
if (millis() - runTime >= 100) { // se ejecuta 500milisegundos
```

```
    runTime = millis();
```

```
    if(tesmod==1){reading = t;}
```

```
}
```

```
int xpos = 0, ypos = 5, gap = 4, radius = 52;
```

```
//Draw a large meter
```

```
xpos = 320/2 - 160, ypos = 0, gap = 100, radius = 105;
```

```
ringMeter(reading,0,100,  xpos,ypos,radius,"Celsius",GREEN2RED); // Draw  
analogue meter
```

```
}
```

```
}
```

```
// Copia cadena de flash a puerto serie
```

```
// ¡La cadena fuente DEBE estar dentro de una declaración PSTR ()!
```

```
void progmemPrint(const char *str) {
```

```
    char c;
```

```
    while(c = pgm_read_byte(str++)) Serial.print(c);
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
}
```

```
void progmemPrintln(const char *str) {
```

```
    progmemPrint(str);
```

```
    Serial.println();
```

```
}
```

```
//
```

```
#####
```

```
#####
```

```
// Dibuja el medidor en pantalla mientras mal alta la temperatura cambia de color
```

```
//
```

```
#####
```

```
#####
```

```
int ringMeter(int value, int vmin, int vmax, int x, int y, int r, char *units, byte scheme)
```

```
{
```

```
// El valor mínimo de r es aproximadamente 52 antes de que el texto de valor se
```

```
// inmiscuya en el anillo , dibujar primero el texto es una opción .
```

```
    x += r; y += r; // Calcular coordenadas del centro del anillo
```

```
    int w = r / 3; // La anchura del anillo exterior es 1/4 de radio
```

```
    int angle = 150; // La mitad del ángulo de barrido del medidor (300 grados)
```

```
    int v = map(value, vmin, vmax, -angle, angle); // Asigne el valor a un ángulo v
```

```
    byte seg = 3; // Los segmentos tienen 3 grados de ancho = 100 segmentos para 300 grados
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
byte inc = 6; // Dibuja segmentos cada 3 grados, aumenta a 6 para el anillo
//segmentado

// Variable para guardar el color del texto "value" del esquema y establecer el
//predeterminado

int colour = BLUE;

// Dibujar bloques de color cada inc grados

for (int i = -angle+inc/2; i < angle-inc/2; i += inc) {

    // Calcule el par de coordenadas para el inicio del segmento

    float sx = cos((i - 90) * 0.0174532925);

    float sy = sin((i - 90) * 0.0174532925);

    uint16_t x0 = sx * (r - w) + x;

    uint16_t y0 = sy * (r - w) + y;

    uint16_t x1 = sx * r + x;

    uint16_t y1 = sy * r + y;

    // Calcular par de coordenadas para el segmento final

    float sx2 = cos((i + seg - 90) * 0.0174532925);

    float sy2 = sin((i + seg - 90) * 0.0174532925);

    int x2 = sx2 * (r - w) + x;

    int y2 = sy2 * (r - w) + y;

    int x3 = sx2 * r + x;

    int y3 = sy2 * r + y;

    if (i < v) { // Rellena los segmentos de colores con 2 triángulos

        switch (scheme) {
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
case 0: colour = RED; break; // color fijo

case 1: colour = GREEN; break; // color fijo

case 2: colour = BLUE; break; // color fijo

case 3: colour = rainbow(map(i, -angle, angle, 0, 127)); break; // Espectro
completo azul a rojo

case 4: colour = rainbow(map(i, -angle, angle, 70, 127)); break; // Verde a rojo
//(alta temperatura, etc.)

case 5: colour = rainbow(map(i, -angle, angle, 127, 63)); break; // Rojo a verde
//(batería baja, etc.)

default: colour = BLUE; break; // color fijo
}

tft.fillTriangle(x0, y0, x1, y1, x2, y2, colour);

tft.fillTriangle(x1, y1, x2, y2, x3, y3, colour);

}

else // Rellena segmentos en blanco

{

tft.fillTriangle(x0, y0, x1, y1, x2, y2, GREY);

tft.fillTriangle(x1, y1, x2, y2, x3, y3, GREY);

}

}

// Convertir valor a una cadena

char buf[10];

byte len = 2; if (value > 999) len = 4;

dtostrf(value, len, 0, buf);
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
buf[len] = ' '; buf[len] = 0; // ¡Añadir espacio de supresión y terminador, también
//ayuda a centrar el texto!

// Establezca el color del texto como predeterminado

tft.setTextSize(1);

if(value>9){

tft.setTextColor(colour,BLACK);

tft.setCursor(x-25,y-10);tft.setTextSize(5);

tft.print(buf);}

if(value<10){

tft.setTextColor(colour,BLACK);

tft.setCursor(x-25,y-10);tft.setTextSize(5);

tft.print(buf);}

tft.setTextColor(WHITE,BLACK);

    tft.setCursor(x-39,y+75);tft.setTextSize(2);

tft.print(units); // Units display

// Calcula y devuelve la coordenada x del lado derecho

return x + r;

}

//

#####

#####

// Devuelve un color de arco iris de 16 bits

//

#####

#####
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
unsigned int rainbow(byte value)
{
// Se espera que el valor esté en el rango 0-127
// El valor se convierte a un color de espectro de 0 = azul a 127 = rojo

byte red = 0; // Red is the top 5 bits of a 16 bit colour value
byte green = 0; // Green is the middle 6 bits
byte blue = 0; // Blue is the bottom 5 bits
byte quadrant = value / 32;

if (quadrant == 0) {
    blue = 31;
    green = 2 * (value % 32);
    red = 0;
}

if (quadrant == 1) {
    blue = 31 - (value % 32);
    green = 63;
    red = 0;
}

if (quadrant == 2) {
    blue = 0;
    green = 63;
```

CAPÍTULO III DESARROLLO DEL SISTEMA DE MONITOREO

```
    red = value % 32;
}

if (quadrant == 3) {
    blue = 0;
    green = 63 - 2 * (value % 32);
    red = 31;
}

return (red << 11) + (green << 5) + blue;
}

//
#####
#####

// Devuelve un valor en el rango -1 a +1 para un ángulo de fase dado en grados

//
#####
#####

float sineWave(int phase) {
    return sin(phase * 0.0174532925);
}
```

Lo siguientes es verificar el programa en el IDE de Arduino que no tenga ningún tipo de error, una vez verificado se procede a cargar el programa en la placa de Arduino Mega para ello el procedimiento es similar al de Arduino UNO, solo que en esta ocasión buscamos en el apartado de placa “Arduino /Genuino Mega or Mega 2560”, después seleccionamos el puerto donde se encuentre conectado, y por ultimo presionamos el botón para cargar que tiene una flecha en la figura 3.19 se puede apreciar los pasos descritos con anterioridad [21,22,23].

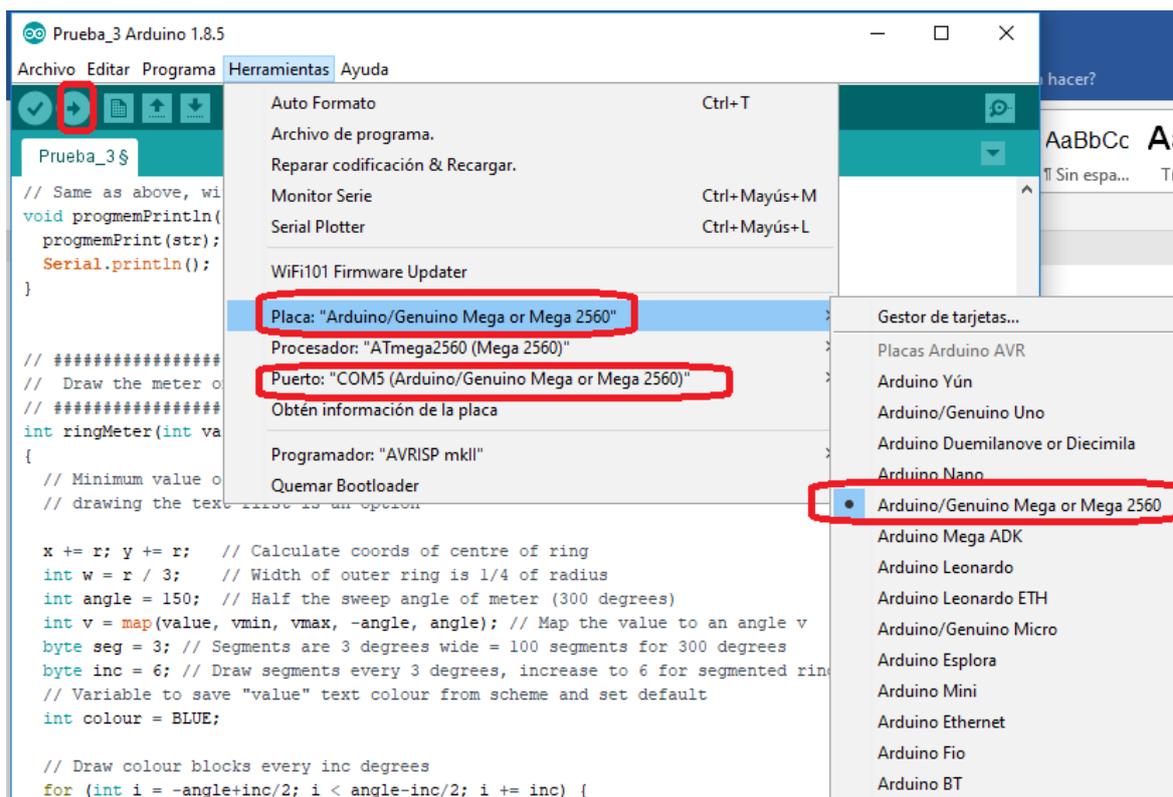


Figura 3.19 Carga de Programa en placa Arduino Mega

3.9 Alimentación de Dispositivos Tanto emisor como Receptor

Para la alimentación se optó por un cargador con las siguientes especificaciones:

Entrada de voltaje: 100-240~0,5ª 50/60Hz

Salida de voltaje:12,0Vcc 1ª

Para verlo físicamente observar figura 2.13.

Esto fue ya que la mayoría de los invernaderos modernos en la actualidad ya cuenta con redes eléctricas que subministran sus áreas de trabajo.

CAPÍTULO IV RESULTADOS

4. Resultados

Los dispositivos finales siendo que el Arduino Mega con la TFT y el Xbee son el Receptor e interpretador de datos visuales sobre el panel gráfico y el Arduino UNO junto con el Shield expensor y el xbee conectados, además de los sensores sería el dispositivo emisor de datos, en la figura 4.1 se distinguen ambos dispositivos.

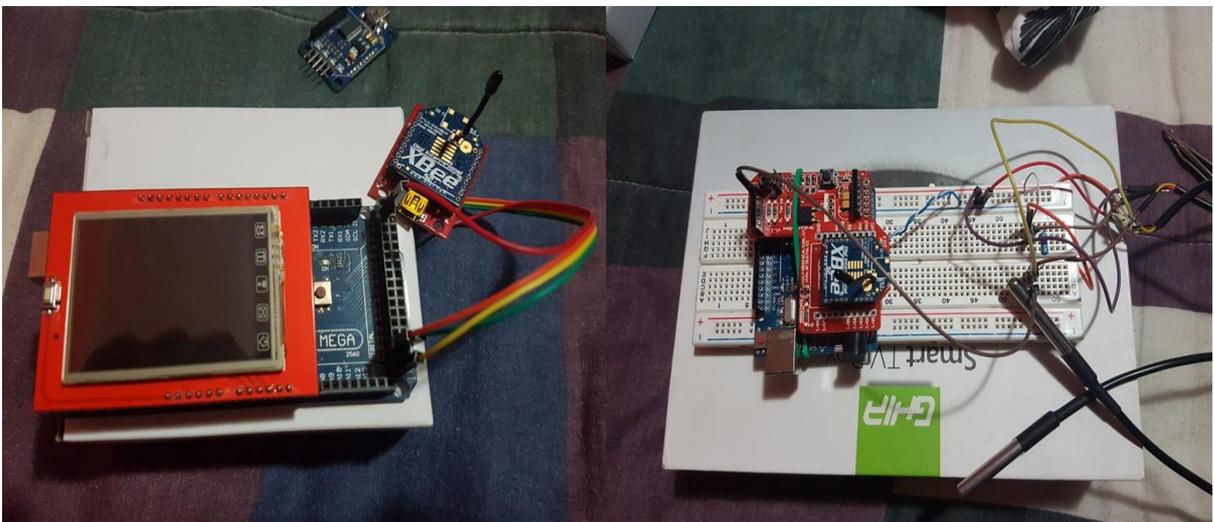


Figura 4.1 Dispositivos Finales

En las siguientes Figuras se puede observar el funcionamiento de ambos dispositivos además del cambio de temperatura, así como la iluminación constante por parte de la pantalla TFT.

En la figura 4.2 y 4.3 se muestra el funcionamiento de los dispositivos tanto receptor como emisor.

CAPÍTULO IV RESULTADOS



Figura 4.2 Funcionamiento Receptor

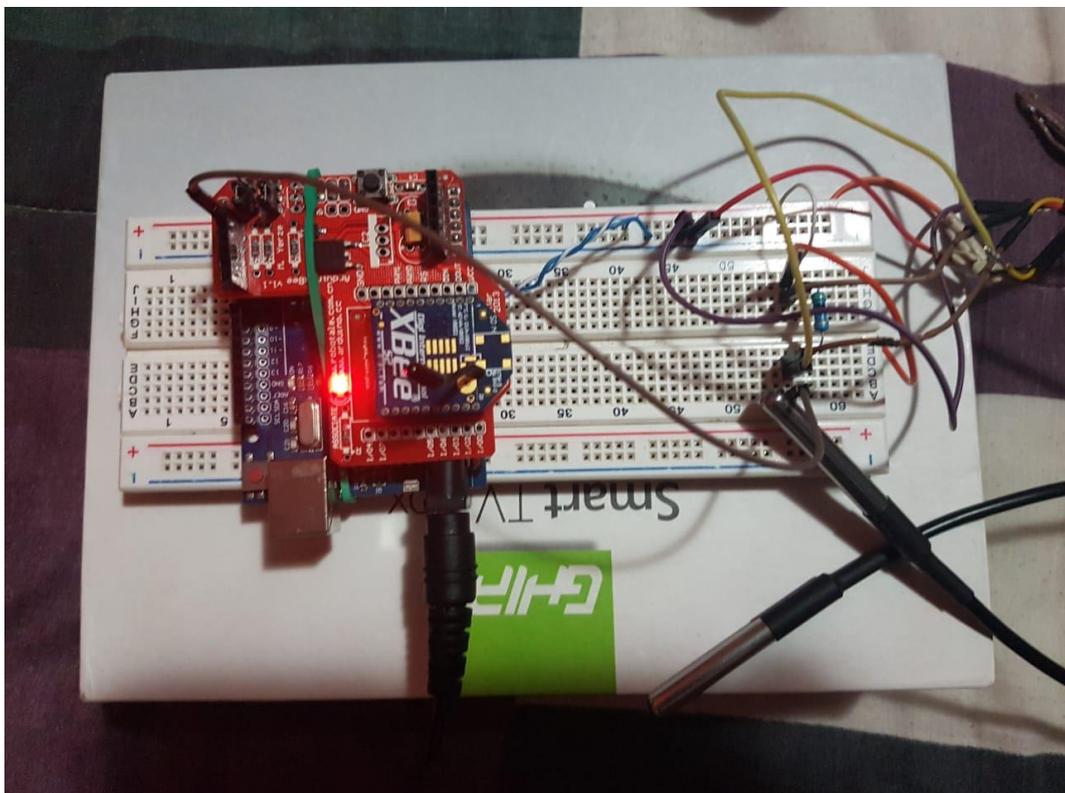


Figura 4.3 Funcionamiento Emisor

CAPÍTULO IV RESULTADOS

En las figuras siguientes se muestra como al elevar la temperatura como es que la pantalla la gráfica constantemente y mientras mas sea la temperatura el color cambia.



Figura 4.4 Cambios de temperatura 1

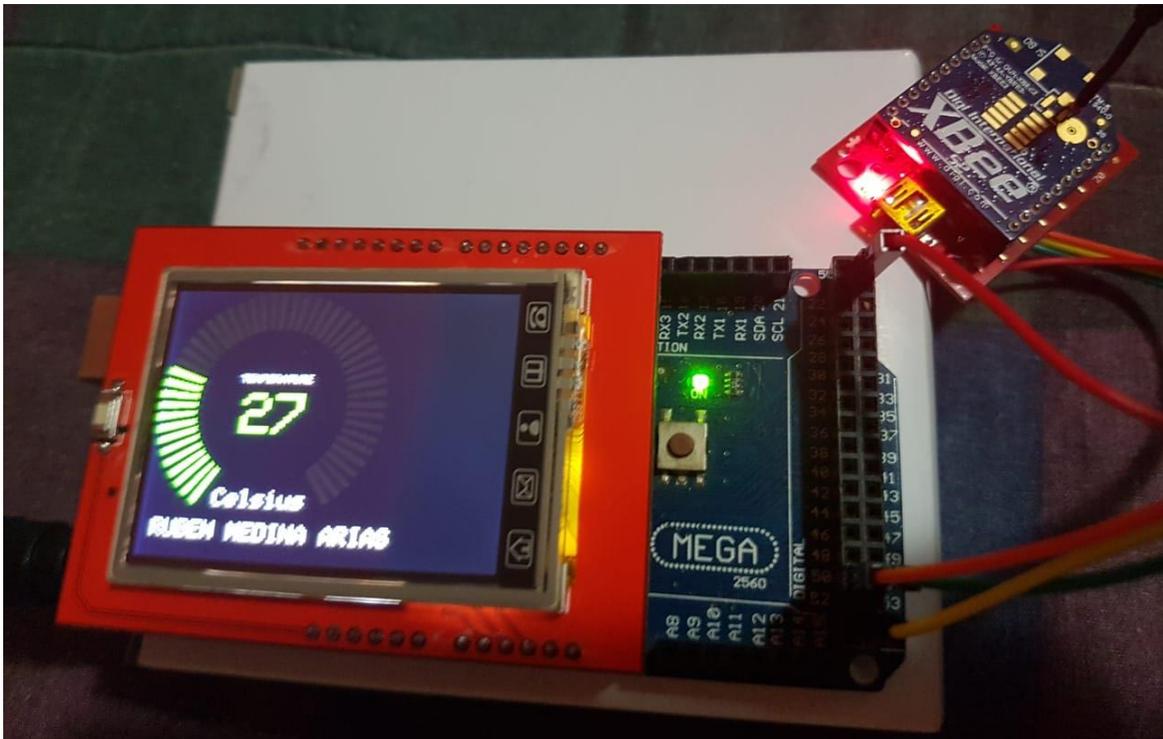


Figura 4.5 Cambios de temperatura 2

CAPÍTULO IV RESULTADOS



Figura 4.6 Cambios de temperatura 3



Figura 4.7 Cambios de temperatura 4

CAPÍTULO IV RESULTADOS

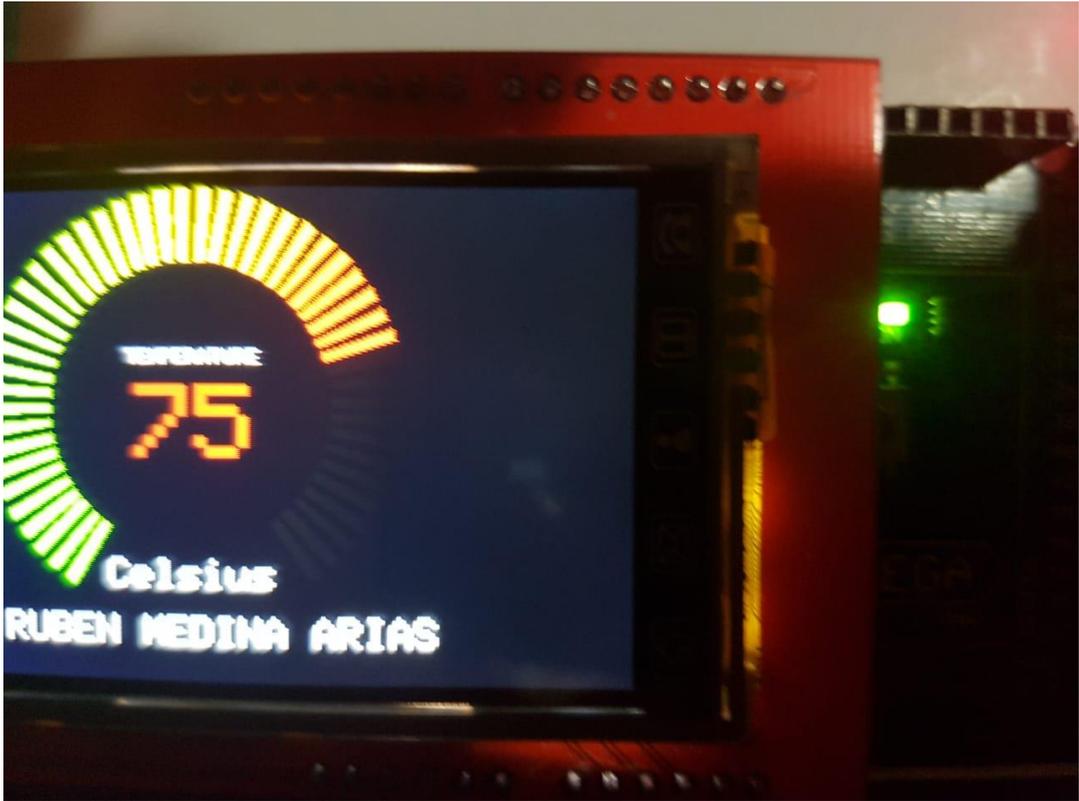


Figura 4.8 Cambios de temperatura 5

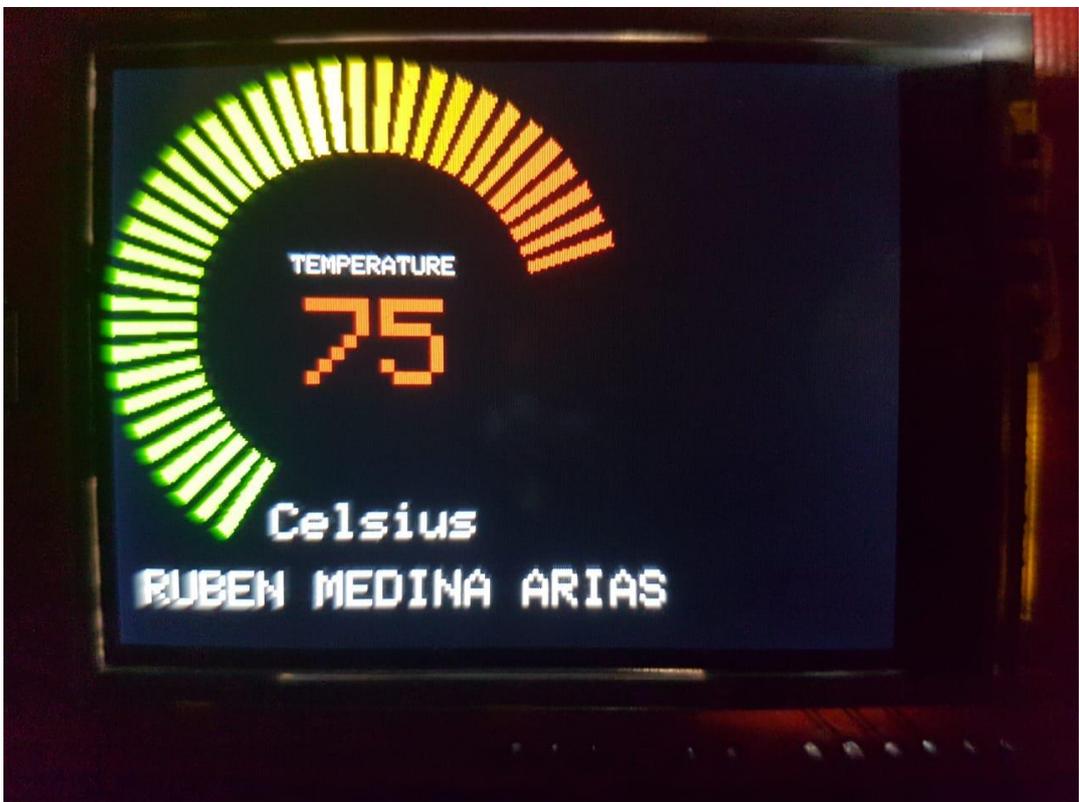


Figura 4.9 Cambios de temperatura 6

CAPÍTULO V CONCLUSIONES

5. Conclusiones

Este trabajo queda con muchas mejoras y posibilidades de crecimiento, por ejemplo, añadir algún control de temperatura automático.

El objetivo principal se alcanzó, el cual era poder adecuar el dispositivo a redes de sensores que ya están implementadas con las mínimas modificaciones para realizar una medición óptima y precisa.

El estudio realizado y la investigación sobre todos los componentes dejan un gran conocimiento y muchas oportunidades de crecimiento para las demás generaciones, las cuales podrán continuar con el proyecto para llevarlo a una etapa más completa y con menos limitantes por ejemplo este dispositivo solo tiene un alcance de 1km línea vista , al haber contacto con paredes árboles y demás se reduce el rango de transmisión por lo tanto la colocación de los dispositivos tiene un rango limitado el cual se tiene que tomar en cuenta para la implementación o necesidades sobre donde se planea implementar.

Otra adecuación sería el recibir de diferentes invernaderos temperaturas y seleccionar por medio de la pantalla táctil cual temperatura monitorear, me queda claro que las mejoras para terminar el producto final pueden ser muchas, y quedo satisfecho con lo que se logro en este gran proyecto.

Referencias bibliográficas

[1] Manejo de temperatura, riego, y más (parte II), Pedro Ponce Cruz, 28 de febrero, 2013

<http://www.hortalizas.com/horticultura-protegida/tu-primer-invernadero-manejo-de-temperatura-riego-y-mas-parte-ii/>

[2] Control climático en Invernaderos, Yolanda López Molina 26/08/2005

<http://www.interempresas.net/Horticola/Articulos/77307-Control-climatico-en-invernaderos.html>.

[3] Soluciones para el análisis del clima en invernadero, : Jorge Flores Velázquez, 14 de marzo, 2017

<http://www.hortalizas.com/cultivos/soluciones-para-el-analisis-del-clima-en-invernadero/>

[4] ¿Cómo funciona lo inalámbrico?

<https://arodriguezr.wordpress.com/como-funciona-lo-inalambrico/>

[5] Áreas de aplicación tecnología Zigbee

<https://sites.google.com/site/ingsoftwares5a/areas-de-aplicacion>

[6] Tipos de dispositivos

<https://sites.google.com/site/ingsoftwares5a/tipos-de-dispositivos>

[7] Ventajas y desventajas de Zigbee

<https://sites.google.com/site/ingsoftwares5a/ventajas-y-desventajas-de-zigbee>

[8] ¿qué es xbee?

<http://xbee.cl/que-es-xbee/>

[9] Xbee y Arduino

<https://www.arduineando.com/xbee-y-arduino/>

[10] Microcontrolador Arduino UNO

<http://arduino.cl/arduino-uno/>

[11] Microcontrolador Arduino Mega 2560

<http://arduino.cl/arduino-mega-2560/>

[12] Sensor

<https://definicion.de/sensor/>

[13] Especificaciones del sensor DS18B20

https://naylampmechatronics.com/blog/46_Tutorial-sensor-de-temperatura-DS18B20.html

[14] Pantalla LCD TFT de 2,4"

<https://electronicavm.wordpress.com/2015/03/05/tft-lcd-touch-2-4-shield-para-arduino-uno/>

[15] XBee Explorer USB

<http://tdrobotica.co/xbee-explorer-usb/651.html>

[16] Modulo Shield Expansi3n XBee para Arduino

<https://moviltronics.com.co/shields-escudos/26-shield-xbee.html>

[17] XCTU

<https://aprendiendoarduino.wordpress.com/tag/xctu/>

[18] IDE Arduino

<https://www.arduino.cc/reference/en/>

[19] Conexi3n Xbee punto a punto

<https://geekytheory.com/tutorial-arduino-comenzando-con-xbee>

[20] Conexi3n de Arduino UNO con Xbee emisor

<https://www.andresduarte.com/wp-content/uploads/2013/06/xbee.png>

[21] librerías IDE

<https://www.arduino.cc/en/Reference/SoftwareSerial>

[22] Comunicación Serial

<https://www.arduino.cc/reference/en/language/functions/communication/serial/available/>

[23] Comunicación i2c

<http://mindstormsyarduino.blogspot.com/2011/03/arduno-y-la-comunicacion-i2c-libreria.html>

Referencias de libros de texto

1.-Taller de Arduino: un enfoque práctico para principiantes, Autor: Germán Tojeiro Calaza Editor: Marcombo, 2014

2.-Gralla, Preston (2007). Cómo funcionan las redes inalámbricas. Anaya Multimedia. ISBN 978-84-415-2068-4.

3.-Engst, Adam (2005). Introducción a las redes inalámbricas. Anaya Multimedia. ISBN 978-84-415-1561-1

4.-Torrente Artero, Óscar. (2013). Arduino. Curso práctico de formación. México D.F.: Alfaomega Grupo Editor, S.A. de C.V.

5.-Acosta Ponce, María Catalina, Tesis Doctoral "Estudio del estándar IEEE 802.15.4 "ZIGBEE" para comunicaciones inalámbricas de área personal de bajo consumo de energía y comparación con el estándar IEEE 802.15.1 "BLUETOOTH"". Escuela Politécnica Nacional de Ecuador, 2006.

6.-El libro de Talk Making Things Talk (Inglés) Pasta blanda – 1 sep 2007 por Tom Igoe (Autor)

7.-El manual del usuario de MaxStream -

http://www.makingthings.com/resources/datasheets/manual_xb_oem-rfmodules_802.15.4.pdf