UNIVERSIDAD NACIONAL DE CHIMBORAZO

FACULTAD DE INGENIERÍA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

Proyecto de Investigación previo a la obtención del título de Ingeniero en Electrónica y Telecomunicaciones

TRABAJO DE TITULACIÓN

Título del proyecto

IMPLEMENTACIÓN DE UN SISTEMA WSN DE MONITOREO EN CONTENEDORES DE BASURA PARA PREVENIR RIESGOS DE INSALUBRIDAD POR ACUMULACIÓN DE DESECHOS SOBRE UNA ZONA URBANA

Autor: Diego Andrés Andrade Segarra

Tutor: Ing. Deysi Inca

Riobamba - Ecuador

Año 2017

Los miembros del Tribunal de Graduación del proyecto de investigación de título: **"IMPLEMENTACIÓN DE UN SISTEMA WSN DE MONITOREO EN CONTENEDORES DE BASURA PARA PREVENIR RIESGOS DE INSALUBRIDAD POR ACUMULACIÓN DE DESECHOS SOBRE UNA ZONA URBANA"** presentado por: **Diego Andrés Andrade Segarra** y dirigida por: la **Ing. Daysi Vilma Inca Balseca.**

Una vez escuchada la defensa oral y revisado el informe final del proyecto de investigación con fines de graduación escrito en la cual se ha constatado el cumplimiento de las observaciones realizadas, remite la presente para uso y custodia en la Biblioteca de la Facultad de Ingeniería de la UNACH.

Para constancia de lo expuesto firman:

Ing. Deysi Inca
Tutora de Tesis

Firma

Ing. Juan Carlos Cepeda
Miembro del Tribunal

Firma

Ing. Alfonso Gunsha
Miembro del Tribunal

Firma

AUTORÍA DE LA INVESTIGACIÓN

La responsabilidad del contenido de este Proyecto de Titulación corresponde exclusivamente a: **Diego Andrés Andrade Segarra e Ing. Daysi Vilma Inca Balseca**; y el patrimonio intelectual de la misma a la Universidad Nacional de Chimborazo.

Diego Andrés Andrade Segarra

CI. 060419973-7

AGRADECIMIENTOS

En primer lugar, agradezco a Dios por haber guiado mi camino al elegir mi profesión; a mis padres, mi más grande apoyo en la vida, velando siempre por mi bienestar y superación.

Agradezco a mi familia, quienes han estado en los momentos más difíciles y en los de mayor regocijo.

A mis amigos, con quienes he compartido buenos y gratos momentos durante este largo trayecto.

Y mi gratitud a aquellas y aquellos docentes que supieron ser verdaderos amigos, pero sobre todo ejemplo de superación y que en su momento supieron apoyarme cuando lo necesité.

DEDICATORIA

Dedico este trabajo a mi familia, especialmente a mi madre, Carmita, que ha sabido dar todo de ella para brindarme apoyo, ánimo, cariño, llenándome de valores y logrando convertirme en la persona que soy. A Galo, que me ha brindado apoyo incondicional de muchas formas, como a un hijo.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Con la utilización de contenedores de basura en distintas ciudades del país como un método de pre-acopio, se solventaron problemas referentes a horarios de recolección y posterior abandono de desechos, pero por otro lado, ahora surge la necesidad de monitorizar los contenedores con el fin de conocer cuando estos se encuentran llenos o saturados de desechos, lo que conlleva a que la población coloque bolsas de basura fuera de los mismos, provocando focos latentes de insalubridad y contaminación. Por ello, se opta por implementar una red de sensores inalámbricos WSN que permitan obtener información de los contenedores referente a que tan llenos se encuentran, gracias a una periódica toma de muestras dentro de los mismos con la ayuda de sensores Sharp de distancia conectados a una modesta batería de LI-PO y a una tarjeta Arduino UNO, se encuentra encargada de la calibración de dicho sensor y posterior conversión y envío de la información obtenida por medio de un dispositivo XBee (End Device) en configuración de red Punto-Multipunto, hacia un módulo WSN compuesto también de un dispositivo XBee (Coordinador), Arduino UNO y una batería LI-PO. Este módulo tiene como tarea recibir los datos de todos los módulos WSN (End-Device) y mostrarlos al usuario monitor por medio de una interfaz simple y comprensible, es por ello que se compone solamente de un display que muestra los estados de todos los contenedores en la red ZigBee, los mismos muestran el número del contenedor (CX) y las palabras: Vacío, normal o lleno.

Abstract

With the use of trash containers in cities across the country like a pre-recollection method, problems related to recollection schedules and subsequent waste abandonment were resolved, but now on the other hand, appear the need of monitory the containers with the purpose to know when they are full or saturated with trash, which means that the user put their trash bags outside them, causing latent sources of insalubrity and pollution. Therefore, it's implement a system of WSN (wireless sensors network) that allow obtain information of the containers related to how full they are, all thanks to a periodic sampling inside them with help of a Sharp distance sensors connected to a LI-PO battery and also to an Arduino UNO card, that is used to control the calibration of the sensors and later convert and send the information obtained by an XBee device (End Device) in Point-Multipoint network configuration, to a other WSN module but also composed of an XBee device (Coordinator), Arduino UNO and a LI-PO battery. The task of this module is receive the data of all WSN (End-Device) modules, and show them to a Monitor User, with the help of an simple and understandable interface, composed only by a display that shows the states of all containers inside the ZigBee network, they show the container number (CX) and the words: empty, normal or full.

Reviewed by: Moyota, Patricia
Language Center Teacher

INTRODUCCIÓN

PROBLEMA

Algunas ciudades del Ecuador; como son: Quito, Guayaquil, Cuenca, Ambato, Riobamba, entre otras, tienen implementado el sistema de contenedores de basura, los cuales son manejados manualmente por los usuarios de los mismos que, no prevén riegos de insalubridad o no conocen especificaciones sobre cuanta capacidad es la óptima para considerar como lleno a un contenedor de basura. Los fabricantes de contenedores ofrecen sus productos con características atractivas para sus potenciales clientes; tomando como punto neural, la responsabilidad en el cuidado del medio ambiente, con un manejo adecuado de los desechos. Estas empresas incursionan en la veta de puntos ecológicos, para la separación de desechos, que promueven y faciliten el reciclaje de los mismos; pero no se ha previsto en su producto, un sistema de monitoreo de contenedores de basura para prevenir riesgos de insalubridad, por acumulación de desechos sobre zonas urbana.

JUSTIFICACIÓN

Como solución al problema, se plantea implementar un sistema WSN (Wireless Sensor Network) ZigBee. Este sistema WSN consiste en una serie de módulos inalámbricos que permiten el intercambio de información entre sí, integrando sensores para supervisar elementos físicos dentro de los contenedores de basura. Con un sistema WSN se mejorará la respuesta del personal encargado de recolección de los desechos alojados en recolectores de basura, ya que unos pueden requerir ser atendidos con mayor prioridad que otros. Con un sistema WSN, se pretende potenciar el uso de las nuevas tecnologías y su aplicación a la solución de un problema, utilizando de dispositivos autónomos con eficientes prestaciones. Es por eso que considerar un sistema WSN ZigBee en contenedores de basura, es proponer una nueva alternativa para prevenir riesgos de insalubridad por acumulación de desechos en zonas urbanas, una nueva alternativa que se proyecta al futuro según las expectativas de todo ser humano, todo ello gracias a las características flexibles tanto por el costo, así como la comunidad mundial que mantiene el continuo desarrollo de este estándar inalámbrico con un sinfín de aplicaciones aún sin descubrir.

OBJETIVOS

General

Implementar un sistema WSN de monitoreo en contenedores de basura.

Específicos

Prevenir riesgos de insalubridad por acumulación de desechos sobre una zona urbana.

Promover la utilización de sensores inalámbricos en aplicaciones de monitoreo.

Probar la eficiencia de dispositivos WSN frente a sensores cableados.

CAPÍTULO I

1 ESTADO DEL ARTE

1.1 ANTECEDENTES

La recolección de basura es una de las técnicas más antiguas y se encarga del manejo de desechos y se utiliza en todas partes del mundo para mantener la salubridad en distintas ciudades, y aunque a lo largo del tiempo la técnica ha ido evolucionando con diversos servicios que facilitan la vida de los usuarios, pero es hasta la actualidad que existen varias ideas aunque sin desarrollo formal sobre sistemas de monitorización de contenedores de basura, con objetivos como el de controlar niveles en tanques de acopio de desechos para la posterior recolección y direccionamiento de dicho material por un sistema de recolección planificado, logrando así conocer donde y cuando se producen desbordamientos o aglomeraciones de desechos que generan una considerable insalubridad y controlar automáticamente el despliegue de vehículos que realizan las tareas de recolección programadas, aunque con técnicas de comunicación no definidas propiamente (YACHAY, 2016). También existen otros diseños de prototipos que integran áreas de electrónica e informática siguiendo metodologías de prototipos evolutivos, que se encargan de dar seguimiento a vehículos de recolección, pero no directamente a los contenedores de basura (Muñoz Rivodó, 2016). El porcentaje de contaminación por acumulación de basura es proporcional a la 4 cantidad de población que se concentra en una zona urbana, datos técnicos ofrecen los niveles de contaminación por ciudad, un ejemplo es la ciudad de Ambato, donde dicho valor es del 30,61%, y la misma tiene un despliegue de contenedores de basura en zonas específicas que ofrecen un 67,56% de cobertura para el servicio (Arias, 2016). La capacidad de las telecomunicaciones en la actualidad, permiten el desarrollo de sistemas de monitoreo en tiempo real y con una gran fiabilidad de la información transmitida y recopilada, ya sea con enlaces simples o aplicaciones más robustas como son trabajos de campo. Monitorizar sistemas que en un principio no fueron concebidos con esa idea, permite aumentar sus capacidades de funcionamiento inicial, mejorando el servicio para lo que fueron ideados y dando posibilidades de futura expansión. La importancia de este proyecto radica en que puede realizar una monitorización por muestras aplicando una transmisión integrada por un mismo tipo de sensores que permiten obtener información

temprana ante eventuales casos donde existan contenedores de basura llenos en una zona urbana, evitando contaminación por la acumulación de desechos y abaratando costos de mantenimiento de este tipo de infraestructuras.

1.2 WIRELESS SENSOR NETWORK (WSN)

Un sistema WSN hace referencia a una red de sensores inalámbricos, con aplicaciones que van desde redes caceras, hasta transmisiones a larga distancia en sistemas de monitoreo continuo en bosques, reservas o ciudades, todo ello sin la necesidad de computadores o servidores de red.

1.3 ZIGBEE

El protocolo ZigBee es una muy popular tecnología utilizada en la creación de redes de radio sensores y es interpretado como un conjunto de protocolos de alto nivel de comunicación inalámbrica. Nace como una solución ante aplicaciones que requieren bajo consumo energético. Cada dispositivo ZigBee dentro de una red WSN, tiene la capacidad de transmitir o recibir información única, por esta razón se considera como un sistema de ayuda para realizar direccionamiento individual de datos, enfatizando que dichas transmisiones poseen un grado de seguridad al ofrecer encriptación de datos de hasta 128bits. Los dispositivos ZigBee son capaces de formar redes entre sí, determinando cual o cuales de estos dispositivos de red se comportarán como routers sin casi intervención de terceros. El mayor desarrollador de este tipo de tecnología es la empresa DIGI International Inc., y comercializa dispositivos bajo este estándar bajo el nombre de XBee, con un sinnúmero de variantes, pero todas ellas cumpliendo con el mismo protocolo de funcionamiento y comunicación (Faludi, 2011).

1.4 ESTÁNDAR IEEE 802.15.4

El estándar IEEE 802.15.4 se encarga de definir niveles físicos y de control de acceso al medio de redes inalámbricas de área personal, con ventajas como bajos costos de producción sin perjuicios de adaptabilidad. Está limitado a bajas tasas de transmisión o LR-WPAN, misma que en su forma más básica puede ofrecer transmisiones de hasta de 10 metros @ 250 kbps, y hasta 1Mbps con rango de 1500 metros en las versiones más

avanzadas. Este estándar, en función a su uso puede operar en una de tres bandas de frecuencia de uso no regulado, las mismas se muestran en la Tabla 1.

Tabla 1. Rangos de Frecuencias y canales XBee

Rango de Frecuencias (Mhz)	Región	No. Canales de comunicación
868 - 868,8	Europa	3
902 -928	Norte América	30
2400 – 2483,5	Mundial	16

***Fuente**: Autor*

1.5 TOPOLOGÍAS DE RED

Un sistema WSN, al estar basado en el estándar IEEE 802.15.4, define dos tipos de nodo para una red (ver Figura 1), donde el primero, es un FFD o dispositivo de funcionalidad completa (full-function device); este puede hacer el papel de coordinador de una red aérea personal PAN o bien comportarse como un nodo normal. FFD básicamente se implementa como un modelo general de comunicación, mismo que permite un intercambio con cualquier otro dispositivo. Contrapuesto a FFD está el segundo tipo de nodo, conocido como RFD o dispositivos de funcionalidad reducida (reduced-function device). Los RFD son sencillo, con recursos y necesidades de comunicación muy limitadas, es por eso que solo pueden comunicarse con FFDs y no pueden hacer el papel de coordinadores. (Ugarte, 2011)

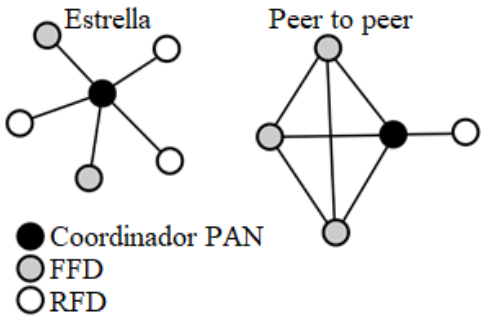


Figura 1. Topologías en Estrella (conexión punto a multipunto) y Peer to peer

***Fuente**: Autor*

1.5.1 CONEXIÓN PUNTO MULTIPUNTO

Este tipo de conexión permite que un mensaje transmitido desde un nodo de red lo reciban solamente las estaciones base o viceversa, pero no ambas al mismo tiempo, es por ello que previamente se identifica la estación receptora desde la dirección de destino de un mensaje. El tipo de conexión punto a multipunto se utiliza en topologías de red en estrella (Ugarte, 2011).

1.6 ÁRBOL DE CLÚSTERS IEEE 802.15.4

Para el estándar IEEE 802.15.4 se puede considerar una estructura en la que se aprovechan los RFDs (reduced-function device), haciendo que estos se conecten exclusivamente con un FFD (full-function device) al mismo tiempo, para así formar redes en las que los RFDs simplemente parte de árbol y la mayoría de nodos son FFDs. Cabe mencionar que dentro de una estructura es posible la creación de redes en mallas genéricas. (José A. Gutiérrez, Edgar H. Callaway Jr., Raymond L. Barrett Jr., 2004)

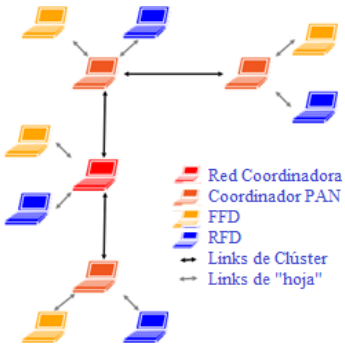


Figura 2. Diagrama de Árbol de Clúster IEEE 802.15.4.

***Fuente**: Autor*

1.7 XBEE

Estos dispositivos operan bajo el estándar 802.15.4. Son capaces de recibir comandos e instrucciones, e incluso leer señales externas gracias a sus entradas/salidas analógicas y digitales, todo ello sin la intervención de un microcontrolador externo, e incluso, por sí solos son capaces de controlar varios dispositivos electrónicos, como LED o buzzers. Su modo de comunicación en la mayoría de sus variantes es gracias una modulación 16QAM

y un radio de 2,4GHz, que juntos logran un excelente alcance de interconexiones. (Faludi, 2011)

1.8 COORDINADOR

Toda red XBee ZigBee requiere de un elemento primordial, y es el tener un módulo coordinador, en el que se deben configurar tres elementos clave (PAIN ID, Destination address high y destination address low) que determinarán el funcionamiento del resto de la red y de futuras configuraciones de nuevos dispositivos que se vayan añadiendo. En sí, es quien recopila e intercambia la información proveniente de toda una red. (Faludi, 2011)

1.9 ROUTER-END DEVICE

Son los dispositivos intermedios o finales de una red XBee ZigBee. Antes que nada, se debe resaltar que una red de este tipo puede o no poseer un dispositivo intermediario o Router, lo que quiere decir que, para tener una red Zigbee funcional, basta con interconectar un dispositivo Coordinador con un dispositivo End-Device. Un dispositivo router o End-Device, es aquel destinado a estar en medio o en un extremo de una red ZigBee como se mencionó anteriormente, donde el router sirve como un nodo de interconexión entre End-Devices. Los dispositivos End-Device son ciertamente los que siempre están destinados a recopilar o recibir información para, o desde el dispositivo coordinador. Normalmente este tipo de dispositivos deben acogerse a la configuración que previamente fue determinada por el módulo coordinador. (Faludi, 2011)

1.10 DIRECCIONAMIENTO PAN

El PAN ID, es básicamente un identificador de red, se utiliza para englobar dispositivos XBee dentro de una misma red y es el primer parámetro en ser configurado y utilizado principalmente en el módulo coordinador. (Faludi, 2011)

1.11 DIRECCIONAMIENT O BÁSICO DH-DL

DH y DL (Destination address high y destination address low) son las dos direcciones físicas y por ende fijas que todo dispositivo XBee posee para formar una red, estas pueden o no ser determinadas por el módulo coordinador, pero sí que son necesarias en modos de

transmisión AT o API. No se debe utilizar estas direcciones si lo que se desea es crear una red de tipo punto a multipunto. Un punto importante es que, todos los dispositivos XBee comparten la misma dirección DH, siendo esta 13A200. (Faludi, 2011)

1.12 MODO TRANSPARENTE AT

Conocido como el modo de transmisión transparente, se utiliza para crear comunicaciones seriales directas entre dos dispositivos XBee. Toda la información enviada con este método es recibida inmediatamente por el destinatario. Para configurar este modo de transmisión son requeridos los parámetros de DH y DL de los dos dispositivos a comunicar (ver Figura 3), y obligatoriamente uno de ellos debe ser el módulo coordinador y el otro puede entrar en modo router o End-Device, ambos módulos deben compartir la misma dirección de PAN ID para formalmente estar dentro de la misma red. (DIGI, DIGI, 2015)

Figura 3. Direccionamiento DH y DL en dispositivos XBee

***Fuente**:Autor*

1.13 AT BROADCAST

El modo AT broadcasts, crea una red punto a multipunto entre un coordinador y varios End-Device o routers. Básicamente replica la información de uno de los módulos y la envía a todos los demás dentro de la red. Este es el modo de comunicación más práctico dentro de los distintos modos de transmisión con ZigBee. No requiere parámetros de DH y DL y es por ellos que estos valores se dejan con ceros, pero sí que necesita de una PAN ID igual en todos los dispositivos para integrarlos dentro de una misma red. Este es el modo de transmisión utilizado para este proyecto en particular. (Chinchu, 2013)

1.14 MODO API

Los dispositivos XBee vienen configurados por defecto en modo AT, por lo que el modo API es un modo de comunicación opcional y de mayores prestaciones que el tradicional AT. Entre sus principales ventajas permite directamente la creación de redes en malla, y su única desventaja el grado de complejidad durante el direccionamiento. El modo API se compone de una estructura compuesta por tramas, como se muestra en la Figura 4. Este modo de transmisión requiere obligatoriamente terminales los suficientemente robustos para enmascarar o desenmascaras las tramas provenientes de este tipo de comunicación. (DIGI, DIGI, 2015)

Start Delimiter	Length		Frame Data								Checksum
1	2	3	4	5	6	7	8	9	...	n	n + 1
0x7E	MSB	LSB	API-specific structure								Single byte

MSB : most-significant byte, **LSB** : least-significant byte

Figura 4. Formato de trama en modo API

***Fuente**: http://docs.digi.com/display/RFKitsCommon/Frame+structure*

1.15 VIDA DE BATERÍA EN XBEE

Una de las principales ventajas y características de los dispositivos XBee es el bajo consumo que poseen. Todo dispositivo XBee posee una configuración Sleep-mode que puede ser aprovechada para ahorrar incluso más energía, especialmente en dispositivos End-Device. El fabricante especifica que un dispositivo XBee, puede funcionar hasta 60 días con un par de baterías convencionales con ciclos de uso de un segundo y descanso o Sleep de 59 segundos. El consumo de corrientes instantáneo en un dispositivo XBee va entre 1uA y 20mA, como se ve reflejado en la Figura 5. (DIGI, DIGI, 2015)

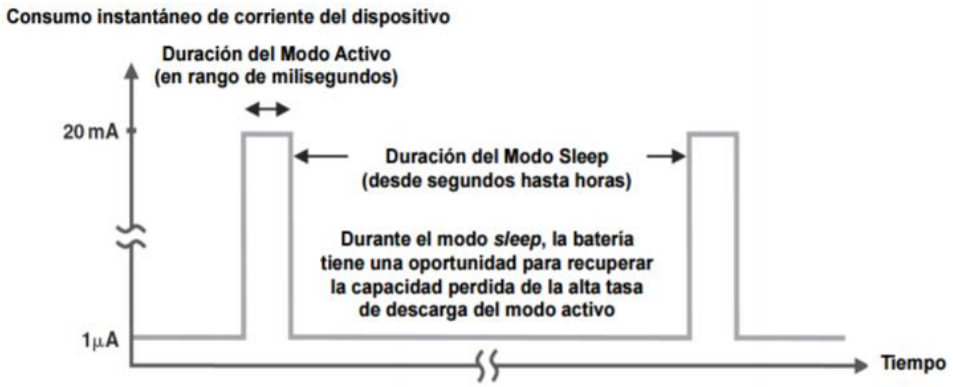


Figura 5. Consumo instantáneo de corriente en módulos XBee

***Fuente**: Autor*

1.16 POTENCIA DE TRANSMISION DE XBEE

La potencia de transmisión en dispositivos Xbee, difiere directamente según el modelo utilizado y del tipo de antena del mismo (SMD, wired, chip). Existen dos modelos de XBee, XBee Series X y XBee Series X PRO (ver Tabla 2). La serie PRO de Xbee son los más robustos en cuanto a alcance y potencia de trasmisión. Este proyecto en particular incluirá en su desarrollo tres dispositivos de la serie Xbee estándar, y un modelo Xbee PRO. (Faludi, 2011).

Tabla 2. Distancias de Funcionamiento XBee

Prestaciones	**Dispositivo XBee**	**Dispositivo XBee PRO**
Alcance en interiores/contexto urbano (m)	30	100
Alcance en exteriores y línea recta (m)	100	1500
Potencia de transmisión(mW)-(dBm)	1-0	100-20
Velocidad de transmisión RF (Kbps)	250	250
Velocidad de datos de la interfaz (pbs)	1200-115200	1200-115200

Voltaje de alimentación (V)	2.8-3.4	2.8-3.4
Frecuencia de alimentación (GHz)	2.4	2.4
Dimensiones	(2.44) X (2.76)	(2.44) X (3.29)

***Fuente**: Autor*

1.17 VELOCIDAD DE TRANSMISIÓN XBEE

La velocidad de transmisión promedio de los dispositivos XBee es algo reducida en comparación con estándares similares, como por ejemplo el Bluetooth o el Wifi, con una tasa de transmisión de máximo de 250Kbps, siendo esta velocidad suficiente para realizar tareas de comunicación en condiciones típicas para dispositivos WSN. (Kurniawan, 2014)

Tabla 3. Comparativa XBee vs. Bluetooth y WiFi

COMPARATIVA	Tasa de bits	Rango Típico	Ejemplo de uso
ZigBee	20 a 250Kbps	10-100m	WSN
Bluetooth	1 a 3 Mbps	2-10m	Wireless Headset
IEEE 802.11b	1 a 11 Mbps	30-100m	Wireless Internet

Fuente: Autor

1.18 XCTU

XCTU es una aplicación libre propiedad de la empresa DIGI International, misma que también fabrica los dispositivos XBee. Mediante su interfaz simple, permite configurar los dispositivos XBee e incluso probarlos mediante, un modo AT integrado en el software, o graficar una red totalmente detallada de una malla ZigBee configurada en modo API. (Faludi, 2011)

1.19 ARDUINO-XBEE SHIELD

La tarjeta de desarrollo bajo lenguaje de programación C, Arduino UNO, mundialmente conocida por su reducido costo y modestas prestaciones, integra un microcontrolador ATMEGA328p, y es elogiada por su amplia integración con un sinfín de accesorios que amplían sus capacidades de interconectividad, como el Shield XBee que permite una fácil integración de esta tarjeta y cualquier dispositivo de la familia XBee. (Faludi, 2011)

1.20 ARDUINO IDE

Es el software con el cual se desarrolla el código para cualquier dispositivo Arduino, su interfaz es sumamente sencilla y posee capacidades de expansión de librerías para casi cualquier dispositivo electrónico, incluidas las requeridas para un correcto funcionamiento entre la integración Arduino UNO+XBee Shield V03+Xbee Series X.

1.21 XBEE EXPLORER

Para poder leer o escribir en un dispositivo XBee, es necesaria una interfaz que lo permita y es precisamente allí donde se requiere de la utilización de un dispositivo Sparkfun XBee Explorer, el mismo sirve como un lector y programador, puede conectarse a cualquier computador con interfaz USB e internamente posee un convertidor de voltaje para reducir los 5V que suministra un computador común en cualquier puerto USB a 3,3V que requiere todo dispositivo XBee. Este dispositivo también puede servir como una interfaz de comunicación serial inalámbrica, que resulta muy útil durante la realización de pruebas de comunicación.

CAPÍTULO II

2 METODOLOGÍA

2.1 CONFIGURACIÓN DE SOFTWARES

2.1.1 CONFIGURACIÓN XCTU

El software XCTU requiere como configuración previa, tener instalados todos los drivers correspondientes a todo el Hardware relacionados con la familia XBee y también del dispositivo Sparkfun XBee Xplorer. Una vez cumplidos los prerrequisitos, se inicia el software por primera vez y, uno de los pasos más importantes, es el de dejar que este automáticamente se actualice por completo, con el fin de tener a disposición las últimas herramientas y características disponibles, pero, sobre todo contar con la última base de datos de firmwares correspondientes a los dispositivos XBee que serán de suma importancia al momento de configurarlos en su momento.

2.1.2 CONFIGURACIÓN ARDUINO IDE

El software Arduino IDE posee en sí mismo los drivers de todo módulo Arduino existente, y es por eso que se requiere que el mismo se encuentre instalado en su última versión disponible. Una vez obtenida e instalada la última versión del software, se requiere de la instalación de las librerías "XBee Arduino library" correspondientes a los módulos XBee para evitar cualquier problema de compilación de toda programación futura relacionada a los mismos, pero, también será necesaria la instalación de otra librería adicional correspondiente a otro elemento contemplado para el desarrollo de este proyecto, la librería "LiquidCrystal", necesaria para dar funcionamiento al código destinado a la interfaz LCD del mismo. Para incluir nuevas librerías en Arduino IDE, se debe seguir la siguiente secuencia: *Programa->Incluir Librería->Gestionar Librerías,* e inmediatamente se desplegará el asistente de librerías. Una vez allí, el software Arduino IDE actualizará automáticamente todas las librerías ya existentes por defecto, y permitirá la búsqueda y adición de nuevas librerías, es entonces cuando se debe escribir en el apartado de búsqueda los nombres de las dos librerías mencionadas anteriormente, y seguido de ello dar clic en instalar para añadirlas a la base de datos.

2.2 CONFIGURACIÓN DE HARDWARE

2.2.1 MÓDULOS WSN END-DEVICE CX

2.2.1.1 CONFIGURACIÓN XBEE SERIES X

Los dispositivos XBee utilizados en los módulos End-Device, compartirán la misma configuración, y únicamente se distinguen por el nombre asignado a cada uno. Para configurar estos dispositivos, primero deben ser conectados al XBee Xplorer (ver Figura 6), y seguido de ello al computador, para a través del software XCTU modificar los parámetros necesarios y así generar una red.

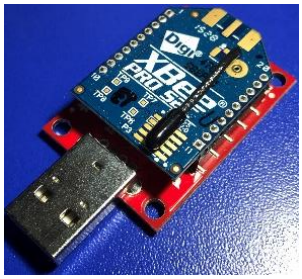


Figura 6. Dispositivo XBee PRO montado en XBee Xplorer

***Fuente**: Autor*

El software XCTU detectará al XBEE Xplorer mediante un puerto COM virtualizado, y una vez leído, permitirá leer o escribir cualquier configuración sobre los dispositivos XBee mediante su interfaz de programación (ver Figura 7).

ID PAN ID	1234	
SC Scan Channels	7FFF	Bitfield
SD Scan Duration	3	exponent
ZS ZigBee Stack Profile	0	
NJ Node Join Time	FF	x 1 sec
NW Network Watchdog Timeout	0	x 1 minute
JV Channel Verification	Disabled [0]	
JN Join Notification	Disabled [0]	
OP Operating PAN ID	1234	
OI Operating 16-bit PAN ID	86E3	
CH Operating Channel	14	
NC Number of Remaining Children	14	
CE Coordinator Enable	Disabled [0]	
DO Device Options	0	Bitfield
DC Device Controls	0	Bitfield

Figura 7. Interfaz de configuración de módulos XBee en XCTU

***Fuente**: Autor*

Para que un dispositivo XBee se convierta en "End-Device", se deben llenar los casilleros del software XCTU como se muestra en la Tabla 4. Cabe mencionar que no todos los casilleros deben ser modificados, por lo que solo deben ser interpretados los descritos en la tabla antes mencionada.

Tabla 4. Configuración en XBees WSN End-Device

PARÁMETRO	**CONFIGURACIÓN/DATO**
ID PAN ID	1234
CE Coordinator Enable	Disable [0]
DH Destination Address High	0
DL Destination Address Low	0
NI Node Identifier	CONTENEDOR X ***(# de contenedor)****
BD Baud Rate	9600 [3]
AP API Mode Enable	Transparent mode [0]
SM Sleep Mode	No Sleep (Router) [0]

***Fuente**: Autor*

2.2.1.2 PROGRAMA EN ARDUINOS END-DEVICE

Está diseñado para que, adquiera los valores correspondientes a la variación de voltaje provenientes de sensores SHARP 2Y0A21-F66 de distancia a través del puerto analógico A0 del módulo Arduino UNO, y luego de iniciar el puerto serial del mismo a 9600 baudios, es necesario realizar una operación matemática que permita la conversión del valor leído por el sensor en Voltios para convertirlo a Centímetros (ver Figura 8.), seguido de ello se da orden al programa para que muestre, por medio del puerto serial, el valor de distancia calculada en esa unidad. Lo siguiente es programar un rango de valores utilizando una

estructura lógica IF/else, y sirve para crear los rangos de valores que el sistema de monitoreo de contenedores de basura utilizará para diferenciar entre los estados ***LLENO, NORMAL o VACÍO***, todo ello como parte de la etapa de calibración de sensores. Los rangos de valores se determinaron en función a datos de distancia vs. voltaje establecidos por el fabricante de los sensores SHARP (refiérase al datatasheet en Anexo 5, pág 55) y depende de la correcta calibración de los mismos. La utilización de lo anteriormente descrito se muestra en la Figura 8.

```
// Pines de lectura
int ir_sensor0 = A0;
void setup()
{
  // inicia comunicaciones serie a 9600 bps
  Serial.begin(9600);


}

void loop()
{
  int lectura, cm;

  lectura = analogRead(ir_sensor0); // lectura del sensor 0
  cm = pow(3027.4 / lectura, 1.2134); // conversión a centímetros
  Serial.print(cm); // lectura del sensor 0
```

Figura 8. Programa de módulos Arduino End-Device CX

***Fuente**: Autor*

2.2.1.3 CALIBRACIÓN DE SENSORES

El fabricante del sensor SHARP establece que con una fuente de alimentación de 5V se tiene un rango de funcionamiento de 10cm a 80cm teóricos para el mismo con su respectivo margen de error (refiérase al datatasheet en Anexo 5, pág 53), lo que en términos de voltaje se traduce a un rango de 5V a 0V teóricos. Pero, como la mayoría de dispositivos de este tipo, los tres sensores de distancias utilizados en el proyecto aun siendo del mismo fabricante y modelo, muestran diferentes rangos de funcionamiento, lo que conlleva a realizar un tratamiento individual para cada uno. Por esta razón, se crea una forma de programación que englobe una configuración de rangos para cualquier sensor, y la misma se determina de la siguiente forma a manera de comparación de rangos individuales con una estructura IF/else: Para comprobar SI (IF) el estado es LLENO, se

realiza una comparativa del valor obtenido en centímetros por el sensor durante una muestra y se la compara con el valor mínimo que tolere el sensor utilizado en ese momento más uno (+1) para determinar si este es MENOR al mismo; para comprobar SI (IF) el estado es VACÍO, se compara el valor obtenido por el sensor en centímetros durante una muestra y se lo compara con el valor máximo que tolere el sensor utilizado menos uno (-1) para determinar si este es MAYOR O IGUAL al mismo; CASO CONTRARIO (ELSE), si los valores leídos por el sensor en centímetros se encuentran fuera de los rangos descritos anteriormente, el estado será NORMAL. Los rangos establecidos se muestran en las Figuras 9 y 10.

```
  if (cm < 6)
  {Serial.print("*C1 LLENO");
}
else if (cm >= 50)
 {Serial.print("*C1 VACIO");
}
else

  {Serial.print("*C1 NORMAL");
}
delay(5000);
}
```

Figura 9. Código de "Estados" para módulos WSN End-Device

***Fuente**: Autor*

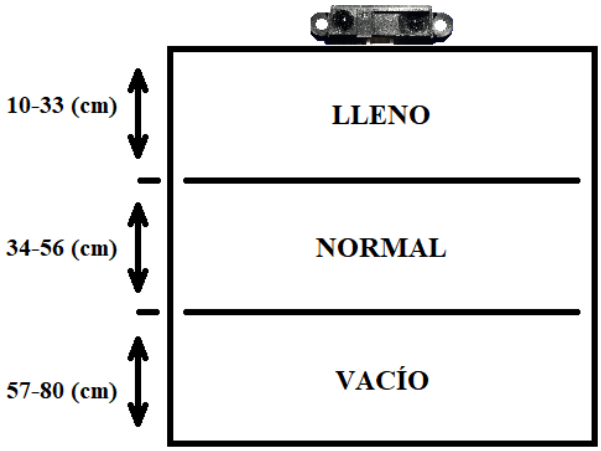


Figura 10. Interpretación de rangos de distancia

***Fuente**: Autor*

2.2.1.4 TIEMPOS DE MUESTREO

Como se puede apreciar en la Figura 9, al final del código en el que se establecen los rangos para cada estado, se encuentra un "delay" con un valor de 5000ms, con ello se establece que, cada cinco segundos se debe realizar una nueva muestra en cada módulo WSN End-Device, pero para evitar colisión de paquetes o saturación en el puerto serial del módulo WSN coordinador, se establece un rango de 1ms en los siguientes módulos WSN End-Device, resultando para estos, un delay de 5000+1ms en el caso del dispositivo C2, o de 5000+1ms para el dispositivo C3.

2.2.1.5 CONFIGURACIÓN XBEE SHIELD

El dispositivo Xbee Shield, tanto en los módulos WSN End-Device, así como en el dispositivo WSN Coordinador, puede tener dos estados que dependerán del modo de transmisión que se elija al momento de enviar información proveniente del puerto RS-232 de la tarjeta Arduino UNO. El primer modo de transmisión se activa al colocar el Switch integrado en el dispositivo Xbee Shield (Figura 11) en la posición "XBEE", lo que hará que toda comunicación serial proveniente del Arduino UNO sea transmitida utilizando el módulo Xbee Series X/PRO. El segundo modo de transmisión se activa al colocar el Switch integrado en el dispositivo Xbee Shield en la posición "USB", y permitirá que la comunicación serial proveniente del Arduino sea transmitida por el puerto USB integrado en el mismo, mas no por el módulo XBee series X/PRO.

Figura 11. Switch selector del dispositivo XBee Shield

***Fuente**: Autor*

2.2.1.6 DIAGRAMA DE CONEXIÓN MÓDULOS WSN END-DEVICE CX

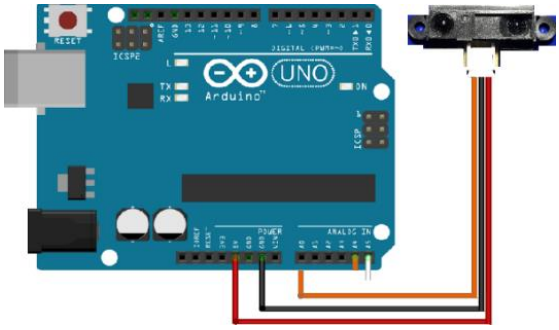


Figura 12. Diagrama de conexión Arduino/Sensor SHARP

***Fuente**: Autor*

Descripción del Diagrama (ver Figura 12):

Los cables Naranja, Negro y Rojo representan Señal, GND y Vcc, y van conectados a la tarjeta Arduino hacia los pines 5V, GND y A0, respectivamente.

El orden de ensamblaje de todos los dispositivos que forman un módulo WSN End-Device se muestra en la Figura 13.

Figura 13. Orden de montaje módulos XBee/XB Shield/Arduino UNO

***Fuente**: Autor*

2.2.2 MÓDULO WSN COORDINADOR

2.2.2.1 CONFIGURACIÓN XBEE PRO

Para configurar el dispositivo XBee del módulo coordinador, primero debe ser conectados al XBee Xplorer (ver Figura 14), y seguido de ello al computador, a través del software XCTU modificar los parámetros necesarios y así generar el nodo principal red. El

software XCTU detectará al XBEE Xplorer mediante un puerto COM virtualizado, y una vez leído, permitirá leer o escribir cualquier configuración sobre el dispositivo XBee PRO.

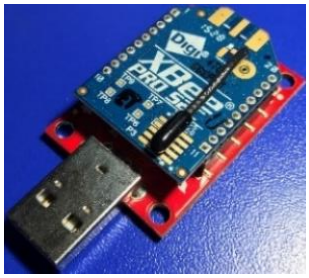


Figura 14. Dispositivo XBee Series X montado en XBee Xplorer

***Fuente**: Autor*

Para que un dispositivo XBee se convierta en "Coordinador", se deben llenar los casilleros del software XCTU como se muestra en la Tabla 5. No todos los casilleros deben ser modificados, solo deben ser interpretados los descritos en la tabla antes mencionada.

Tabla 5. Configuración en XBee WSN Coordinador

PARÁMETRO	**CONFIGURACIÓN/DATO**
ID PAN ID	1234
CE Coordinator Enable	Enabled [1]
DH Destination Address High	0
DL Destination Address Low	FFFF
NI Node Identifier	MONITOR/COORDINADOR
BD Baud Rate	9600 [3]
AP API Mode Enable	Transparent mode [0]
SM Sleep Mode	No Sleep (Router) [0]

***Fuente**: Autor*

2.2.2.2 PROGRAMACIÓN EN ARDUINO COORDINADOR

El programa para Arduino UNO destinado al módulo WSN Coordinador, cumple dos tareas, la primera es leer el puerto serial por el cual vendrán los distintos posibles estados provenientes de cada uno de los tres dispositivos WSN End-Device; la segunda tarea es mostrar la información correspondiente a los estados e imprimirla en un LCD de 16x2. Cabe mencionar que la información se irá mostrando en el LCD de 16x2 píxeles conforme los datos siguen llegando al puerto serial del módulo WSN coordinador, cumpliendo los tiempos de muestreo descritos en el apartado 2.2.1.4. El código empleado para utilizar el LCD de 16x2 píxeles requiere del uso de la librería LyquidCrystal y se muestra en la Figura 15.

```
#include <LiquidCrystal.h>
//Inicializar la librería con los números de los pines de la interface
LiquidCrystal lcd(7, 8, 9, 10, 11, 12); //RS, E, D4, D5, D6, D7
int pinLed = 13;
void setup() {
// Configurando el número de columnas y filas de la LCD
lcd.begin(16, 2);
// Inicializar la comunicación serial
Serial.begin(9600);
// Se define el pin conectado al LED como salida
pinMode(pinLed, OUTPUT);
}
void loop() {
//Proceso de parpadeo del LED
digitalWrite(pinLed, HIGH);
delay(500);
digitalWrite(pinLed, LOW);
delay(500);
// Cuando los caractéres llegan al puerto serial
if (Serial.available()) {
// Borrar la pantalla
lcd.clear();
// Leer todos los caractéres disponibles
while (Serial.available() > 0) {
// Mostrar el mensaje LCD
lcd.write(Serial.read());
}
}
}
```

Figura 15. Programa en Arduino para el módulo WSN Coordinador

***Fuente**: Autor*

2.2.2.3 INTERFAZ LCD

Es la interfaz mediante la cual el usuario monitor podrá monitorizar cualquier tipo de alerta generada por uno o varios de los módulos WSN End-Device, mostrando los mensajes previamente programados en estos módulos con el formato CX LLENO, CX NORMAL o CX VACÍO. El LCD irá actualizando la información de su interfaz conforme

vayan llegando las alertas, cumpliendo con los tiempos de muestreo descritos en el apartado 2.2.1.4. Para la interfaz se utiliza un display LCD de 16x2 píxeles, su diagrama de conexión se aprecia en el apartado 2.2.2.4, Figura 16.

2.2.2.4 DIAGRAMA DE CONEXIÓN MÓDULO WSN COORDINADOR

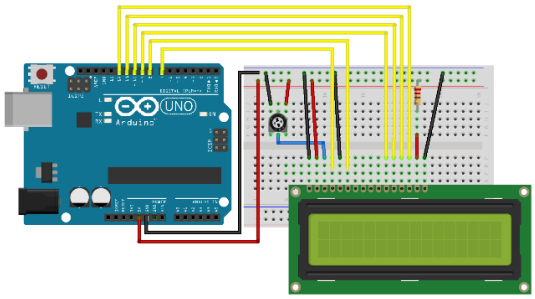


Figura 16. Diagrama de conexión módulo WSN Coordinador

***Fuente**: Autor*

Descripción del diagrama (ver Figura 16):

Los cables amarillos se interconectan desde las salidas 7, 8, 9, 10, 11 y 12 de la tarjeta Arduino hacia los pines RS, E, D4, D5, D6 y D7 respectivamente.

Los cables Rojo y Negro representan Vcc y GND, respectivamente.

La resistencia de 220ohm se utiliza para la conexión del pin de iluminación del LCD.

El potenciómetro de 1Kohm se utiliza para regular el contraste del LCD.

2.2.3 DIMENSIONAMIENTO DE BATERÍAS

Para sustentar tanto los módulos WSN End-Device y el módulo coordinador, se contemplaron baterías de LI-PO de 750mAh, con una capacidad de descarga de 25C @3,7V. Un dispositivo XBee promedio consume 0.12mAh (MaxStream, 2005), lo que se traduce a requerir una batería de 1216.4mAh para un año de funcionamiento, es decir, un 61,6% de esa proyección será conseguida con la batería descrita al principio de este apartado, lo que se traduce en más de medio año de vida de batería por carga.

$$\frac{750mAh * 100\%}{1216{,}34} = 61{,}66\%$$

Ecuación 1. Proyección de batería XBee

***Fuente**: (MaxStream, 2005)*

2.2.4 DIAGRAMA DE RED WSN

La red en topología Punto-multipunto se genera a partir de la configuración del hardware en modo de transmisión broadcast. El diseño de red contemplado en topología punto-multipunto (estrella), se muestra en la Figura 17.

Figura 17. Diagrama de red WSN en estrella (punto-multipunto)

***Fuente**: Autor*

Descripción del diagrama:

1. Módulo WSN coordinador.
2. Transmisión ZigBee en topología punto-multipunto (estrella).
3. Módulos WSN End-Device CX en contenedores de basura.

La disposición de red está ideada para que la conexión ZigBee trabaje en broadcast. Este modo de funcionamiento permitirá que, cualquier transmisión realizada por los módulos WSN End-Device, sea replicada hacia los demás módulos WSN presentes en la misma red, pero al final solo el módulo WSN coordinador será el único capaz de interpretar todas las transmisiones en la red, mientras que los demás módulos no realizaran acción alguna ante cualquier recepción de información, de allí proviene la interpretación de conexión "punto-multipunto". Particularmente se escogió utilizar este tipo de topología porque a nivel de configuración es mucho más sencillo de aplicar a la práctica. Cabe mencionar que el hardware utilizado es apto para trabajar en otros modos de transmisión y no solo con el citado anteriormente, por ejemplo, es capaz de funcionar con topología en malla, que resulta mucho más robusta que una topología en estrella, con la diferencia de que a nivel

de configuración es bastante más compleja de llevar a la práctica sin la ayuda de hardware más complejo, como con un computador.

2.2.5 TRANSMISIÓN BROADCAST

Como se mencionó varias veces en el apartado 2.2.4, la comunicación para enviar los datos desde los módulos WSN End-Device hasta los módulos WSN Coordinador está configurada en Broadcast, lo que permite que la transmisión proveniente de cualquier WSN End-Device se refleje "teóricamente" en todo dispositivo dentro de la misma red incluyendo otros End-Device, pero el único capaz de interpretar los datos recibidos es el WSN Coordinador, y es capaz de realizar dicha tarea porque tanto la configuración mostrada en la Tabla 5 para el dispositivo XBee PRO, como la del código de Arduino en la Figura 15 así lo permiten, mientras que los otros códigos y configuraciones de XBee y Arduino para los WSN End-Device se limitan enviar información, mas no leerla.

2.2.6 DIAGRAMA POSICIONAMIENTO DE MÓDULOS WSN

2.2.6.1 POSICIONAMIENTO WSN END-DEVICE

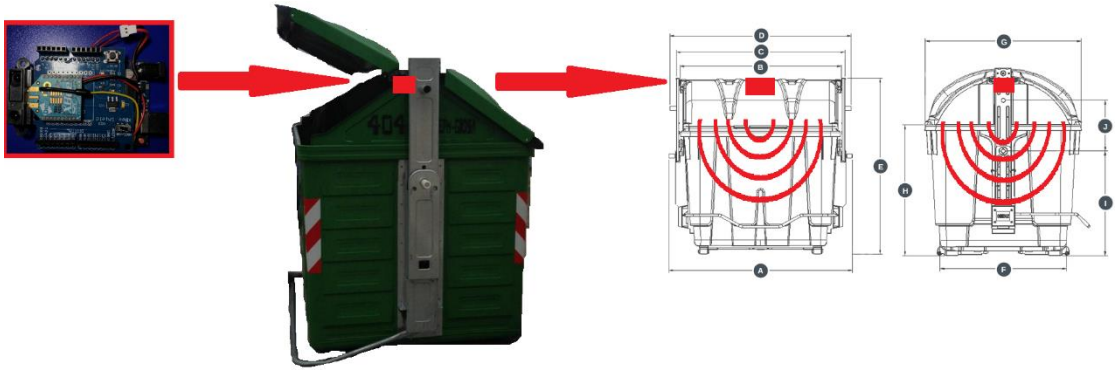


Figura 18. Diagrama de posicionamiento módulos WSN End-Device

***Fuente**: Autor*

Descripción del diagrama (ver Figura 18):

Los módulos WSN End-Device se montan en la parte superior central interna de los contenedores que se desean monitorear. Se debe considerar que, la posición del sensor Sharp de distancia tiene que quedar dispuesto hacia abajo, con el fin de poder sensar que distancia existe entre este y la basura en el interior de un contenedor.

2.2.6.2 POSICIONAMIENTO WSN COORDINADOR

Figura 19. Diagrama de montaje módulo WSN Coordinador

***Fuente**: Autor*

Descripción del diagrama (ver Figura 19):

El módulo WSN coordinador debe ser colocado vehículos de recolección de basura, su posición debe ser un lugar cómodo de visualizar por el usuario encargado de vigilar los distintos estados de cada contenedor.

2.2.6.3 FUNCIONAMIENTO DEL SISTEMA WSN

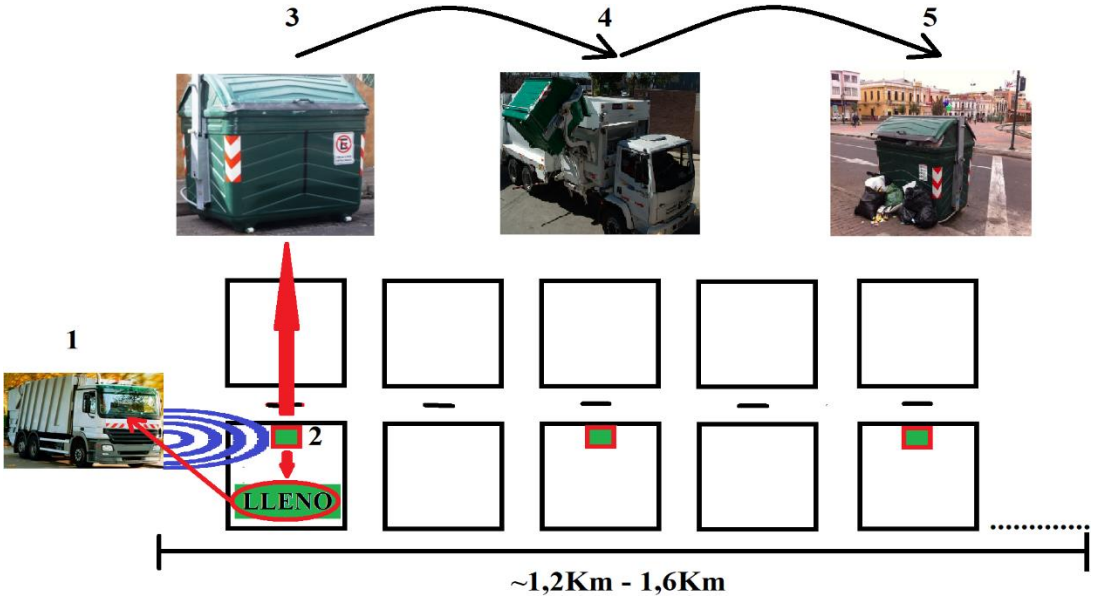


Figura 20. Diagrama de funcionamiento Sistema de monitoreo WSN

***Fuente**: Autor*

Descripción del Diagrama (ver Figura 20):

1. El vehículo recolector de basura, gracias a su Sistema WSN de monitoreo de contenedores, es capaz de visualizar teóricamente, todos los contenedores en un radio de aproximadamente 1,6Km, y el mismo sigue aumentando conforme el vehículo avanza en su recorrido.

2. Se determina que contenedores de basura están llenos durante el trayecto de un camión de basura, y este puede dar prioridad a los mismos.

3. El camión se dirige donde se encuentra el contenedor lleno.

4. Recoge todos los desperdicios dentro del contenedor.

5. El sistema evita que, los usuarios de un contenedor lleno, a consecuencia de ello comiencen a generar contaminación por acumulación de desechos sobre una zona urbana.

CAPÍTULO III

3 RESULTADOS Y DISCUSIÓN

3.1 RESULTADOS

3.1.1 PROCESAMIENTO DE LA INFORMACIÓN

El módulo WSN coordinador es el encargado de recibir el estado de todo módulo WSN End-Device integrado a la red del mismo, y para mostrar la información de manera correcta, utiliza un tiempo de separación de 1ms entre las transmisiones de cada módulo WSN End-Device como se explicó en el apartado 2.2.1.4, lo que previene pérdidas de paquetes y sus datos por colisiones de los mismos. El resultado es una transmisión continua e ininterrumpida. Por otra parte, la información de los sensores Sharp analógicos, se procesa de manera correcta en la etapa de conversión de unidades de voltaje a centímetros como parte de la programación estipulada para cada módulo WSN End-Device, mismo valor resultante de la conversión se aprecia en la Figura 21.

Figura 21. Muestra de valores convertidos de (V) a (cm)

***Fuente**: Autor*

3.1.2 PRUEBAS DE COMUNICACIÓN

Las pruebas de comunicación de cada módulo WSN End-Device, se realizaron tanto en un computador mediante la herramienta de Arduino IDE “Monitor Serie” configurado a una velocidad de 9600 baudios, así como también en el propio módulo WSN coordinador;

en ambas técnicas de prueba de funcionamiento se logró un correcto desempeño de comunicación de cada uno de los módulos.

3.1.2.1 TIEMPOS DE CONEXIÓN

La velocidad promedio de conexión (de diez muestras) entre todos los módulos WSN es de ~1 segundo, aunque teóricamente es instantánea o menor a un segundo. En la Tabla 6. se observa el valor promedio del tiempo medido al realizar las pruebas de conexión.

Tabla 6. Tiempos de conexión red WSN

PARÁMETROS	CONEXIONES MAYORES A 1seg.	CONEXIONES MENORES A 1seg.	CONEXIONES IGUALES A 1seg.
NÚMERO DE MUESTRAS	3	3	4

***Fuente**: Autor*

3.1.2.2 ATENUACIONES

Como todo dispositivo de transmisión inalámbrica, un módulo XBee no está exento de atenuaciones o pérdidas ocasionadas por el medio u objetos en el mismo, por eso que se debe considerar la tabla de distancias en el apartado 1.16, la cual da una idea de cuanta distancia disminuye si se utiliza los módulos en ambientes externos o internos, pero al estar enfocado este proyecto a un uso externo, teóricamente se deben alcanzar distancias un 75% cercanas a la máxima teórica de funcionamiento.

3.1.2.3 DISTANCIAS DE FUNCIONAMIENTO DE LA RED

Como se mencionó en el apartado anterior, las atenuaciones deterioran el valor teórico de distancia en el que pueden funcionar los módulos XBee, reduciendo este aproximadamente a un 75%, en la Tabla 7. se muestran las distancias promedio de funcionamiento aplicando dicha reducción de distancia por atenuaciones del medio (el promedio se obtuvo de una prueba de 10 muestras).

Tabla 7. Distancias de funcionamiento

DESCRIPCIÓN	Distancia teórica (m)	Distancia real (m)
XBee PRO	100	75
XBee Series X	1500	1125

***Fuente**: Autor*

En base a las distancias mostradas en la Tabla 7, se determina que la distancia máxima par que exista conexión entre el módulo WSN Coordinador y uno de los módulos WSN End-Device es la suma entre el valor de distancia real tanto del XBee PRO del módulo coordinador, como la del módulo End-Device, lo que daría como resultado un valor máximo de 1200 metros de funcionamiento para la red WSN.

3.1.2.4 POTENCIA DE FUNCIONAMIENTO

Como se describe en el apartado 3.1.2.3, la distancia de trabajo de la red WSN decrementa aproximadamente un 75% por factores atenuadores aledaños a la misma, y ese decremento también es directamente proporcional a una reducción de la potencia de transmisión que llegará hasta cada uno de los módulos WSN al establecer un enlace de red, misma reducción de potencia se describe en la Tabla 8.

Tabla 8. Potencia teórica vs. Real

PARÁMETROS	Potencia Teórica (mW)	Potencia Real (mW)
XBee Series X	3,1	2,33
XBee PRO	63	47,3

***Fuente**: Autor*

3.1.3 INTERFAZ DE USUARIO

Una vez enlazada toda la red WSN, se puede comprobar la interfaz de monitorización incluida en el módulo WSN coordinador, como se aprecia en la Figura 22, haciendo posible el poder leer los distintos estados de cada uno de los dispositivos WSN End-Device.

Figura 22. Interfaz de usuario

***Fuente**: Autor*

3.1.4 MÓDULOS WSN END-DEVICE PROTOTIPO

Este apartado muestra el resultado de consolidar todas las configuraciones, programaciones y dispositivos destinados a formar los módulos WSN End-Device, dicho resultado se aprecia en la Figura 23.

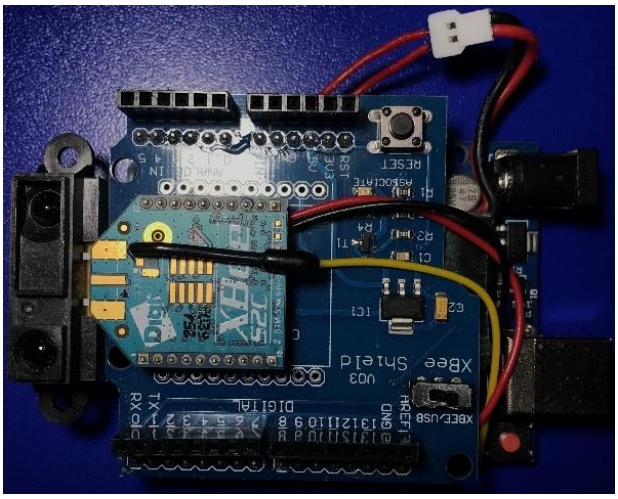


Figura 23. Prototipo de módulo WSN End-Device

***Fuente**: Autor*

3.1.5 MÓDULO WSN COORDINADOR PROTOTIPO

Este apartado muestra el resultado de consolidar todas las configuraciones, programación y dispositivos destinados a formar el módulo WSN Coordinador, dicho resultado se aprecia en la Figura 24.

Figura 24. Prototipo de módulo WSN Coordinador

***Fuente**: Autor*

3.1.6 PRUEBAS Y RESULTADOS

3.1.6.1 ANÁLISIS DE DATOS

Para este apartado, se muestra una comparativa entre el rendimiento de una red en malla y una en estrella, contemplando en ambos casos la misma cantidad de dispositivos XBee de un mismo tipo (XBee series X) y considerando que la disposición de los contenedores de basura se encuentra distribuida sobre una zona urbana en forma lineal como se describe en el apartado 2.2.6.3. Referirse a la Figura 25 para una mejor interpretación.

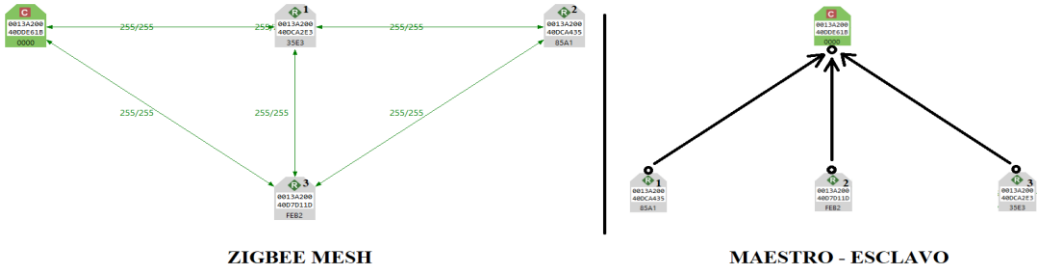


Figura 25. Comparativa ZigBee Mesh vs. Maestro-esclavo

***Fuente**: Autor*

Se considera para cada dispositivo de la red los parámetros de la Tabla 9.

Tabla 9. Distancias máximas según el tipo de red

CONFIGURACIÓN DE RED	**Distancia teórica por dispositivo (m)**	**Potencia teórica por dispositivo (mW)**	**Alcance máximo (m)**
ZigBee Mesh (XBee series X)	100	3.1	600
Maestro-Esclavo (Punto Multipunto, XBee series X)	100	63	200

***Fuente**: Autor*

3.1.6.2 ANÁLISIS ESTADÍSTICO

Para realizar el análisis se recurre a los valores reflejados en la pestaña "Alcance máximo" de la Tabla 9.

Tabla 10. Análisis estadístico usando Chi Cuadrado

CHI CALCULADO

	r	5
	k	2
Grados de Libertad	(r-1)(k-1)	4
Nivel de significación	α	0,0001
ji Cuadrado	X2	23,51
Probabilidad	P	0,00
X2Prueba		50,00

***Fuente**: Autor*

Como se muestra en la Tabla 10, el valor de X2Prueba es mayor al de ju Cuadrado, comprobando que esta investigación obtuvo buenos resultados.

3.1.6.3 WSN VS. SENSORES CABLEADOS

Una red inalámbrica por lo general tiene varias limitaciones, y la que más destaca directamente es la distancia de funcionamiento de los mismos, por lo que en escenarios

donde se plantea extender sensores a lo largo de grandes áreas geográficas siempre se ha preferido optar por sensores cableados, pero con dispositivos WSN se solventa en gran medida estas limitaciones. En base a los resultados obtenidos con este proyecto, se pueden citar una seria de ventajas de utilizar WSN, las mismas se pueden apreciar en la Tabla 10.

Tabla 11. Comparativa WSN vs. Sensores cableados

WSN	**SENSORES CABLEADOS**
Presencia nula de cables para la comunicación, directamente menor inversión de recursos y dinero.	Requiere el uso de cables para establecer una conexión o comunicación.
Permiten crear movilidad y un costo de mantenimiento considerablemente más bajo.	No permite movilidad de sensores.
El tiempo de despliegue e instalación es rápido por el hecho de no requerir cables.	El tiempo de instalación de una red lleva bastante tiempo y requiere equipos para realizarlo.
Permite ir añadiendo componentes adicionales a la red rápidamente.	Permite la adición de nuevos elementos a la red, pero toman tiempo instalar nuevos cables a la misma.
Consumo de energía eficiente.	Consume más recursos energéticos.
Permite transmitir a lugares poco accesibles.	Dificulta realizar instalaciones a lugares muy remotos o poco accesibles.

***Fuente**: Autor*

3.2 DISCUSIÓN

La mayoría de proyectos realizados utilizando dispositivos XBee junto con tarjeas Arduino no son del todo concluyentes, es decir, para mostrar el funcionamiento de una red apenas y utilizan un par de XBee, los mismos se configuran en modo AT y efectivamente establecen una comunicación entre ellos, pero es solo eso. Muchos proyectos investigados como soporte para la realización de esta tesis no son lo suficientemente claros o capaces de explicar o configurar más de dos dispositivos XBee,

lo que en principio dificultó bastante la realización de la investigación y del proyecto en sí. Es por ello que nace la necesidad de indagar más allá, y la esencia de configuración requerida para lograr construir este proyecto la plasmó alguien como respuesta en un foro. Una de las desventajas quizás de utilizar dispositivos XBee es que el modo AT (Transparent Mode) es bastante limitado, pero a su vez muy sencillo de utilizar, y de allí parte el que muchos usuarios lo tomen como su modo de transmisión favorito, sin considerar sus muchas falencias. Como se mencionó en el apartado 1.14, existe el modo de transmisión API, que cubre prácticamente todos los huecos que deja el modo AT, y para empezar permite el crear mallas completas con dispositivos XBee, pero al mismo tiempo es sumamente compleja de manipular. Crear una malla con XBee´s en modo API es sumamente sencillo, e incluso transmitir en ese modo también lo es si se utiliza un computador para realizar esa tarea; el problema radica cuando se desea utilizar hardware más compacto para realizar tareas de procesamiento de datos y transmisión de los mismos, como por ejemplo un módulo Arduino, ya que dentro de la programación se debe establecer una trama, y para ello se requieren varias librerías que tampoco están pulidas del todo. Pero dejando todo eso de lado, utilizar dispositivos XBee ya sea en modo AT o en modo API (con la ayuda de un computador) trae muchas ventajas, es universalizar el uso de sensores a nivel inalámbrico, con practicante nulas pérdidas o problemas de seguridad puesto que se utilice el modo de transmisión que se desee, los XBee siempre encriptan la información que transmiten al medio. Pero principalmente los dispositivos XBee ofrecen una sorprendente autonomía en cuanto al consumo energético, a la vez que son capaces de alcanzar distancias sumamente altas a relación del tamaño que poseen sus antenas, 100m para las versiones estándar, o 1500m en las variantes PRO y una vez más todo ello comparado con el consumo de los mismos. Las aplicaciones con módulos XBee son prácticamente ilimitadas, pueden utilizarse con prácticamente cualquier sensor o actuador electrónicos disponibles en el mercado, lo que los convierte en versátiles herramientas de comunicación a medianas o altas distancias.

CAPÍTULO IV

4 CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

- Utilizar módulos WSN elimina la necesidad de utilizar cables a la hora de desplegar sensores para cualquier aplicación, más aún donde la distancia y la cantidad de los mismos son factores cruciales, lo que al final se traduce en un menor gasto ante la implementación de un proyecto.
- Este proyecto permite tomar muestras de dos modos, directamente en un computador, o a través de un microcontrolador, en este caso un Arduino, sin dejar de ser, en cualquiera de los casos citados un módulo WSN coordinador.
- Los módulos WSN diseñados permiten obtener la información de cada contenedor de basura de forma continua (en tiempo real) o a manera de muestras programadas para que se lleven a cabo según las necesidades del usuario, y a la vez ahorrar batería.
- Los sensores pueden sufrir fallas de calibración y por ende obtener medidas erróneas si la vida útil de las baterías utilizadas comienzan a decaer; a menor cantidad de batería, menor rango de sensado.
- El espacio de 1ms empleado entre tiempos de transmisión para evitar colisión de paquetes, demostró ser suficiente como para mantener toda la red funcionando prácticamente al mismo tiempo, y teóricamente el usuario monitor no debe notarlo.
- Este proyecto genera soporte para la creación de cualquier otro basado en lectura de múltiples sensores a distancia, ya que posee todo lo necesario para añadir cuanto dispositivo se desee a una misma red.
- De ser implementado este proyecto en alguna zona urbana, fomentará la disminución de contaminación por acumulación de desechos.
- El alcance del Sistema de monitoreo WSN sería más robusto si se utilizan únicamente dispositivos XBee de la serie PRO, aumentando el rango de alcance hasta 3Km teóricos en configuración Punto-Multipunto.

4.2 RECOMENDACIONES

- Para un correcto funcionamiento de los sensores, se debe mantener un óptimo nivel de batería, por ello se recomienda revisar el nivel de las mismas cada 4 o 5 meses de uso.
- Tomar en cuenta que al aplicar los módulos WSN en contenedores de basura, estos pueden sufrir daños o incluso los sensores pueden quedar obstruidos por algún tipo de desecho, por lo que se recomienda hacer revisiones periódicas para asegurar la integridad de los dispositivos.
- Mantener los módulos en algún tipo de caja o contendor que ayuden a los mismos a permanecer aislados de los desechos o de usuarios de los contenedores que intencional o inintencionalmente puedan estropearlos.
- Capacitar al personal que esté destinado a estar a cargo de los contenedores de basura para que sepan que protocolo se debe seguir si se detecta algún tipo de inconveniente con uno de los módulos ocasionado por algún tipo de obstrucción de los mismos.
- Si se desea añadir más módulos WSN a este proyecto, simplemente se deben seguir los pasos de programación y configuración descritos en los apartados 2.2.1.1 y 2.2.1.2.
- Para programar un módulo WSN ensamblado por completo, se debe recordar que la posición del switch integrado en el XBee Shield debe estar en posición "USB", e igualmente cuando quiera ser utilizado nuevamente el switch debe regresar a la posición de "XBEE".
- Se recomienda a Universidades y sobre todo a Docentes a fomentar e investigar sobre los dispositivos XBee, especialmente sobre el modo de transmisión API por su infinidad de ventajas antes el modo AT.
- Se puede probar la creación de una malla en modo API si se configuran todos los XBee en modo API y se deja en blanco las configuraciones de DH y DL, solo se requiere una PAN ID en el módulo principal y las misma en los módulos End-Device.

REFERENCIAS BIBLIOGRÁFICAS

Arias, P. &. (12 de Febrero de 2016). *Ecuador en Cifras*. Obtenido de http://www.ecuadorencifras.gob.ec/documentos/webinec/Encuestas_Ambientales/Hogares_2014/Documento_tecnico_Modulo_Ambi

Chinchu. (28 de Enero de 2013). *Forum arduino*. Obtenido de http://forum.arduino.cc/index.php?topic=145253.0

DIGI. (06 de Agosto de 2015). *DIGI*. Obtenido de http://docs.digi.com/display/RFKitsCommon/XBee+transparent+mode

DIGI. (06 de Agosto de 2015). *DIGI*. Obtenido de http://docs.digi.com/display/RFKitsCommon/XBee+API+mode

DIGI. (06 de Agosto de 2015). *DIGI*. Obtenido de http://docs.digi.com/display/RFKitsCommon/Battery+life

Faludi, R. (2011). *Wireless Sensor Networks.* Sebastopol: O´REILLY.

José A. Gutiérrez, Edgar H. Callaway Jr., Raymond L. Barrett Jr. (2004). *Low'Rate Wireless Personal Area Networks.* New York: Standards Information Network IEEE Press.

Kurniawan, A. (2014). *XBee IEEE 802.15.4 Programming.* Berlin: PE Press.

MaxStream. (03 de Noviembre de 2005). *MaxStream*. Obtenido de ftp1.digi.com/support/utilities/batterylifecalculator.xls

MILLIGAN, T. A. (2005). *MODERN ANTENNA DESIGN*. New Jersy: John Wiley & Sons.

Muñoz Rivodó, G. (12 de Febrero de 2016). *USB (Universidad Simón Bolivar*. Obtenido de http://159.90.80.55/tesis/000135543.pdf

Ugarte, D. d. (2011). *Trilogía de las Redes.* España: Grupo Cooperativo de las Indias.

YACHAY. (12 de Febrero de 2016). *YACHAY*. Obtenido de http://www.yachay.gob.ec/wp-content/uploads/downloads/2014/09/Recoleccioninteligente-FICHA.pdf

ANEXOS

ANEXO 1

A continuación, se muestra el código implementado en los dispositivos WSN End-Device:

```
// Pines de lectura
int ir_sensor0 = A0;
void setup()
{  // inicia comunicaciones serie a 9600 bps
  Serial.begin(9600);}
void loop()
{int lectura, cm;
  lectura = analogRead(ir_sensor0); // lectura del sensor 0
  cm = pow(3027.4 / lectura, 1.2134); // conversión a centímetros
  Serial.print(cm); // lectura del sensor 0
   if (cm < 6)
  {Serial.print("*C1 LLENO");}
else if (cm >= 50)
 {Serial.print("*C1 VACIO");}
else
  {Serial.print("*C1 NORMAL");}
delay(5000);}
```

ANEXO 2

En este apartado se muestra el código utilizado en el módulo WSN Coordinador:

```
#include <LiquidCrystal.h>
      //Inicializar la librería con los números de los pines de la interface y definición del pin para el LED
            LiquidCrystal lcd(7, 8, 9, 10, 11, 12); //RS, E, D4, D5, D6, D7
            int pinLed = 13;
void setup() {
            // Configurando el número de columnas y filas de la LCD
            lcd.begin(16, 2);
           // Inicializar la comunicación serial
           Serial.begin(9600);
          // Se define el pin conectado al LED como salida
          pinMode(pinLed, OUTPUT); }
void loop() {
         //Proceso de parpadeo del LED
         digitalWrite(pinLed, HIGH);
         delay(500);
         digitalWrite(pinLed, LOW);
         delay(500);
        // Cuando los caractéres llegan al puerto serial
        if (Serial.available()) {
        // Borrar la pantalla
        lcd.clear();
        // Leer todos los caractéres disponibles
        while (Serial.available() > 0) {
        // Mostrar el mensaje LCD
        lcd.write(Serial.read());
        } } }
```

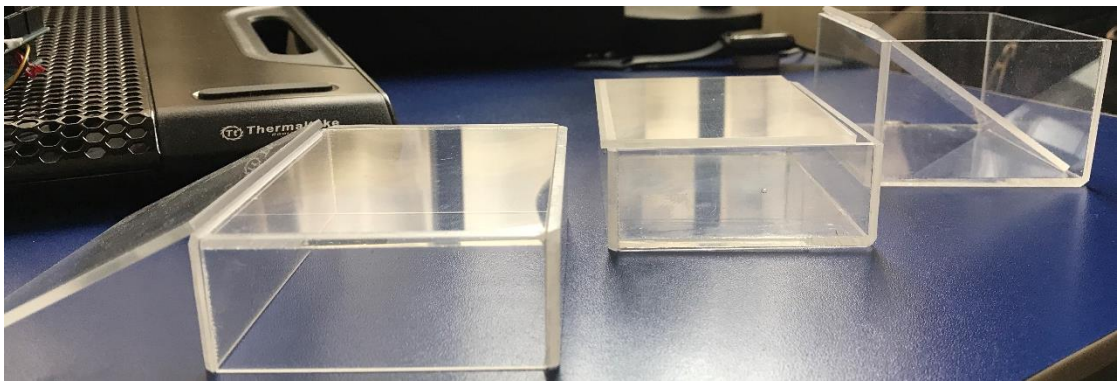
ANEXO 3

Interfaz del programa Battery Life Calcuator V2.2 de ©MaxStream, Inc.

Design Goals							
System efficiency		90%	90%	90%	90%	90%	90%
Target battery life	yr	2	2	2	2	2	2
Required battery capacity	mAh	**2432,75**	**1066,74**	**0,00**	**0,00**	**0,00**	**0,00**
or							
Given battery capacity	mAh	2000	2000	2000	2000	2000	2000
Estimated battery life	yr	**1,64**	**3,75**	**0,00**	**0,00**	**0,00**	**0,00**

ANEXO 4

Cajas de protección para los módulos WSN End-Device y Coordinador



Desde la izquierda, la primera y segunda caja corresponde a los módulos WSN End-Device. Sus dimensiones son: 3,5x7,5x10,5 (cm) (alto x ancho x profundidad). El tercer modelo corresponde a la caja protectora del módulo WSN Coordinador, sus dimensiones son: 5,5x7,5x9,5 (cm) (alto x ancho x profundidad).

ANEXO 5

En este apartado se exponen los Datasheets correspondiente a los dispositivos utilizados para la elaboración del proyecto:

XBEE S2/PRO

DIGI

EMBEDDED RF
MODULES FOR OEMS

XBEE® S2C 802.15.4 RF MODULES

Low-cost, easy-to-deploy modules provide critical end-point connectivity to devices and sensors

XBee RF modules provide OEMs with a common footprint shared by multiple platforms, including multipoint and ZigBee/Mesh topologies, and both 2.4 GHz and 900 MHz solutions. OEMs deploying the XBee can substitute one XBee for another, depending upon dynamic application needs, with minimal development, reduced risk and shorter time-to-market.

XBee 802.15.4 RF modules are ideal for applications requiring low latency and predictable communication timing. Providing quick, robust communication in point-to-point, peer-to-peer, and multipoint/star configurations, XBee 802.15.4 products enable robust end-point connectivity with ease. Whether deployed as a pure cable replacement for simple serial communication, or as part of a more complex hub-and-spoke network of sensors, XBee 802.15.4 RF modules maximize performance and ease of development.

XBee 802.15.4 modules seamlessly interface with compatible gateways, device adapters and range extenders, providing developers with true beyond-the-horizon connectivity.

The updated XBee S2C 802.15.4 module is built with the SiliconLabs EM357 SoC and offers improved power consumption, support for over-the-air firmware updates, and provides an upgrade path to DigiMesh® or ZigBee® mesh protocols if desired.

BENEFITS

- Simple, out-of-the-box RF communications, no configuration needed
- Point-to-multipoint network topology
- 2.4 GHz for worldwide deployment
- Common XBee footprint for a variety of RF modules
- Industry leading sleep current of sub 1uA
- Firmware upgrades via UART, SPI or over the air
- Migratable to DigiMesh and ZigBee PRO protocols and vice-versa

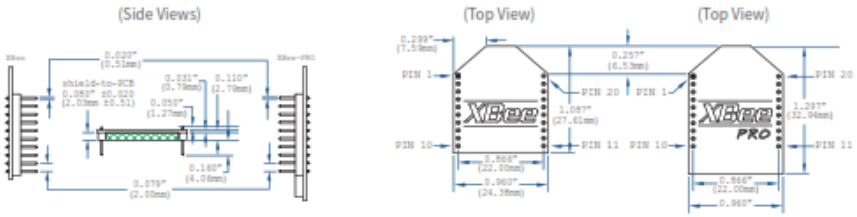
RELATED PRODUCTS

ConnectPort® X4/X4H Gateways | XBee® Adapters | XCTU | Digi Device Cloud™ | Development Kits

APPLICATION EXAMPLE

SPECIFICATIONS	XBee® S2C 802.15.4	XBee-PRO® S2C 802.15.4
PERFORMANCE		
TRANSCEIVER CHIPSET	Silicon Labs EM357 SoC	
DATA RATE	RF 250 Kbps, Serial up to 1 Mbps	
INDOOR/URBAN RANGE	200 ft (60 m)	300 ft (90 m)
OUTDOOR/RF LINE-OF-SIGHT RANGE	4000 ft (1200 m)	2 miles (3200 m)
TRANSMIT POWER	3.1 mW (+5 dBm) / 6.3 mW (+8 dBm) boost mode	63 mW (+18 dBm)
RECEIVER SENSITIVITY (1% PER)	-100 dBm / -102 dBm boost mode	-101 dBm
FEATURES		
SERIAL DATA INTERFACE	UART, SPI	
CONFIGURATION METHOD	API or AT commands, local or over-the-air (OTA)	
FREQUENCY BAND	ISM 2.4 GHz	
FORM FACTOR	Through-Hole, Surface Mount	
HARDWARE	S2C	
ADC INPUTS	(4) 10-bit ADC inputs	
DIGITAL I/O	15	
ANTENNA OPTIONS	Through-Hole: PCB Antenna, U.FL Connector, RPSMA Connector, or Integrated Wire SMT: RF Pad, PCB Antenna, or U.FL Connector	
OPERATING TEMPERATURE	-40° C to +85° C	
DIMENSIONS (L X W X H) AND WEIGHT	Through-Hole: 0.960 x 1.087 in (2.438 x 2.761 cm) SMT: 0.866 x 1.33 x 0.120 in (2.199 x 3.4 x 0.305 cm)	Through-Hole: 0.960 x 1.297 in (2.438 x 3.294 cm) SMT: 0.866 x 1.33 x 0.120 in (2.199 x 3.4 x 0.305 cm)
NETWORKING AND SECURITY		
PROTOCOL	XBee 802.15.4 (Proprietary 802.15.4)	
UPDATABLE TO DIGIMESH PROTOCOL	Yes	
UPDATABLE TO ZIGBEE PROTOCOL	Yes	
INTERFERENCE IMMUNITY	DSSS (Direct Sequence Spread Spectrum)	
ENCRYPTION	128-bit AES	
RELIABLE PACKET DELIVERY	Retries/Acknowledgements	
IDS	PAN ID and addresses, cluster IDs and endpoints (optional)	
CHANNELS	16 channels	15 channels
POWER REQUIREMENTS		
SUPPLY VOLTAGE	2.1 to 3.6V	2.7 to 3.6V
TRANSMIT CURRENT	33 mA @ 3.3 VDC / 45 mA boost mode	120 mA @ 3.3 VDC
RECEIVE CURRENT	28 mA @ 3.3 VDC / 31 mA boost mode	31 mA @ 3.3 VDC
POWER-DOWN CURRENT	<1 µA @ 25° C	<1 µA @ 25° C
REGULATORY APPROVALS		
FCC, IC (NORTH AMERICA)	Yes	Yes
ETSI (EUROPE)	Yes	No
RCM (AUSTRALIA AND NEW ZEALAND)	No (Coming soon)	No (Coming soon)
TELEC (JAPAN)	No (Coming soon)	No (Coming soon)

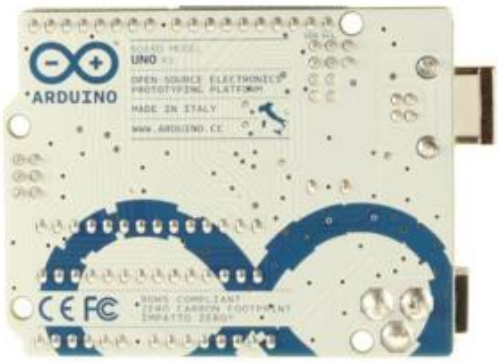
PART NUMBERS	DESCRIPTION
KIT	
XKB2-A2T-WWC	Wireless Connectivity Kit with XBee 802.15.4 (S2C)
MODULES	
XB24CAWIT-001	XBee 802.15.4 through-hole module w/ wire antenna
XB24CAPIT-001	XBee 802.15.4 through-hole module w/ PCB antenna
XB24CAUIT-001	XBee 802.15.4 through-hole module w/ U.fl connector
XB24CASIT-001	XBee 802.15.4 through-hole module w/ RPSMA connector
XB24CAPIS-001	XBee 802.15.4 SMT module w/ PCB antenna
XB24CAUIS-001	XBee 802.15.4 SMT module w/ U.fl connector
XB24CARIS-001	XBee 802.15.4 SMT module w/ RF Pad connector
XBP24CAWIT-001	XBee-PRO 802.15.4 through-hole module w/ wire antenna
XBP24CAPIT-001	XBee-PRO 802.15.4 through-hole module w/ PCB antenna
XBP24CAUIT-001	XBee-PRO 802.15.4 through-hole module w/ U.fl connector
XBP24CASIT-001	XBee-PRO 802.15.4 through-hole module w/ RPSMA connector
XBP24CAPIS-001	XBee-PRO 802.15.4 SMT module w/ PCB antenna
XBP24CAUIS-001	XBee-PRO 802.15.4 SMT module w/ U.fl connector
XBP24CARIS-001	XBee-PRO 802.15.4 SMT module w/ RF Pad connector



ARDUINO UNO

Arduino Uno

Arduino Uno R3 Front



Arduino Uno R3 Back

Arduino Uno R2 Front

Arduino Uno SMD

Arduino Uno Front

Arduino Uno Back

Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.
The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.
Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.
Revision 3 of the board has the following new features:

- 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V

Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Schematic & Reference Design

EAGLE files: arduino-uno-Rev3-reference-design.zip (NOTE: works with Eagle 6.0 and newer)
Schematic: arduino-uno-Rev3-schematic.pdf
Note: The Arduino reference design can use an Atmega8, 168, or 328, Current models use an ATmega328, but an Atmega8 is shown in the schematic for reference. The pin configuration is identical on all three processors.

Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.
External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.
The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.
The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.**This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.

- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the Wire library.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).
A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.
The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.
The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).
You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.
The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

XBEE SHIELD V03

Xbee Shield

NB : This page refers to a deprecated product. Information on the current Wireless Shield can be found here

Overview

The Xbee shield allows an Arduino board to communicate wirelessly using Zigbee. It is based on the Xbee module from MaxStream. The module can communicate up to 100 feet indoors or 300 feet outdoors (with line-of-sight). It can be used as a serial/usb replacement or you can put it into a command mode and configure it for a variety of broadcast and mesh networking options. The shields breaks out each of the Xbee's pins to a through-hole solder pad. It also provides female pin headers for use of digital pins 2 to 7 and the analog inputs, which are covered by the shield (digital pins 8 to 13 are not obstructed by the shield, so you can use the headers on the board itself).
The Xbee shield was created in collaboration with Libelium, who developed it for use in their SquidBee motes (used for creating sensor networks).

Schematic

XbeeShieldSchematic.pdf (Eagle schematics and board layouts available from the Libelium SquidBee wiki download page.)

Jumper Settings

The Xbee shield has two jumpers (the small removable plastic sleeves that each fit onto two of the three pins labelled Xbee/USB). These determine how the Xbee's serial communication connects to the serial communication between the microcontroller (ATmega8 or ATmega168) and FTDI USB-to-serial chip on the Arduino board.
With the jumpers in the **Xbee** position (i.e. on the two pins towards the interior of the board), the DOUT pin of the Xbee module is connected to the RX pin of the microcontroller; and DIN is connected to TX. Note that the RX and TX pins of the microcontroller are still connected to the TX and RX pins (respectively) of the FTDI chip - data sent from the microcontroller will be transmitted to the computer via USB as well as being sent wirelessly by the Xbee module. The microcontroller, however, will only be able to receive data from the Xbee module, not over USB from the computer.
With the jumpers in the **USB** position (i.e. on the two pins nearest the edge of the board), the DOUT pin the Xbee module is connected to the RX pin of the *FTDI chip*, and DIN on the Xbee module is connected to the TX pin of the FTDI chip. This means that the Xbee module can communicate directly with the computer - however, *this only works if the microcontroller has been removed from the Arduino board*. If the microcontroller is left in the Arduino board, it will be able to talk to the computer normally via USB, but neither the computer nor the microcontroller will be able to talk to the Xbee module.

Networking

The Arduino XBee shield can be used with different XBee modules. The instructions below are for the **XBee 802.15.4 modules** (sometimes called "Series 1" to distinguish them from the Series 2 modules, although "Series 1" doesn't appear in the official name or product description).

Addressing

There are multiple parameters that need to be configured correctly for two modules to talk to each other (although with the default settings, all modules should be able to talk to each other). They need to be on the same network, as set by the **ID** parameter (see "Configuration" below for more details on the parameters). The modules need to be on the same channel, as set by the **CH** parameter. Finally, a module's destination address (**DH** and **DL** parameters) determine which modules on its network and channel will receive the data it transmits. This can happen in a few ways:

If a module's **DH** is 0 and its **DL** is less than 0xFFFF (i.e. 16 bits), data transmitted by that module will be received by any module whose 16-bit address **MY** parameter equals **DL**.
If **DH** is 0 and **DL** equals 0xFFFF, the module's transmissions will be received by all modules.

If **DH** is non-zero or **DL** is greater than 0xFFFF, the transmission will only be received by the module whose serial number equals the transmitting module's destination address (i.e. whose **SH** equals the transmitting module's **DH** and whose **SL** equals its **DL**).

Again, this address matching will only happen between modules on the same network and channel. If two modules are on different networks or channels, they can't communicate regardless of their addresses.

Configuration

Here are some of the more useful parameters for configuring your Xbee module. For step-by-step instructions on reading and writing them, see the guide to the Xbee shield. Make sure to prepend `AT` to the parameter name when sending a command to the module (e.g. to read the `ID` parameter, you should send the command `ATID`).

Command	*Description*	*Valid Values*	*Default Value*
`ID`	The network ID of the Xbee module.	0 - 0xFFFF	3332
`CH`	The channel of the Xbee module.	0x0B - 0x1A	0X0C
`SH` and `SL`	The serial number of the Xbee module (`SH` gives the high 32 bits, `SL` the low 32 bits). Read-only.	0 - 0xFFFFFFFF (for both `SH` and `SL`)	different for each module
`MY`	The 16-bit address of the module.	0 - 0xFFFF	0
`DH` and `DL`	The destination address for wireless communication (`DH` is the high 32 bits, `DL` the low 32).	0 - 0xFFFFFFFF (for both `DH` and `DL`)	0 (for both `DH` and `DL`)
`BD`	The baud rate used for serial communication with the Arduino board or computer.	0 (1200 bps) 1 (2400 bps) 2 (4800 bps) 3 (9600 bps) 4 (19200 bps) 5 (38400 bps) 6 (57600 bps) 7 (115200 bps)	3 (9600 baud)

Note: although the valid and default values in the table above are written with a prefix of "0x" (to indicate that they are hexadecimal numbers), the module will not include the "0x" when reporting the value of a parameter, and you should omit it when setting values.
Here are a couple more useful commands for configuring the Xbee module (you'll need to prepend `AT` to these too).

Command	*Description*
`RE`	Restore factory default settings (note that like parameter changes, this is not permanent unless followed by the `WR` command).
`WR`	Write newly configured parameter values to non-volatile (long-term) storage. Otherwise, they will only last until the module loses power.
`CN`	Exit command mode now. (If you don't send any commands to the module for a few seconds, command mode will timeout and exit even without a `CN` command.)

For more details on configuring the Xbee module, see the product manual from MaxStream.

SHARP 2Y0A21-F66

GP2Y0A21YK0F

Distance Measuring Sensor Unit
Measuring distance: 10 to 80 cm
Analog output type

■Description

GP2Y0A21YK0F is a distance measuring sensor unit, composed of an integrated combination of PSD (position sensitive detector) , IRED (infrared emitting diode) and signal processing circuit.
The variety of the reflectivity of the object, the environmental temperature and the operating duration are not influenced easily to the distance detection because of adopting the triangulation method.
This device outputs the voltage corresponding to the detection distance. So this sensor can also be used as a proximity sensor.

■Features

1. Distance measuring range : 10 to 80 cm
2. Analog output type
3. Package size : 29.5×13×13.5 mm
4. Consumption current : Typ. 30 mA
5. Supply voltage : 4.5 to 5.5 V

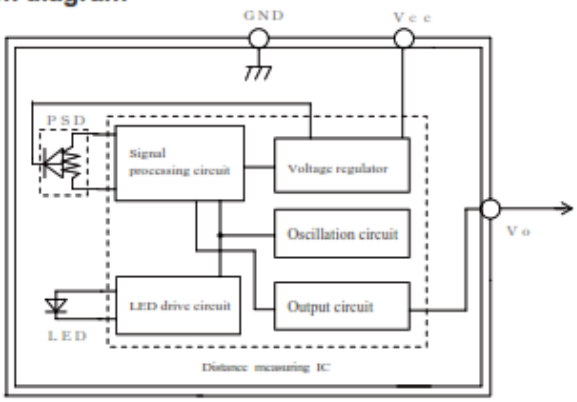
■Agency approvals/Compliance

1. Compliant with RoHS directive (2002/95/EC)

■Applications

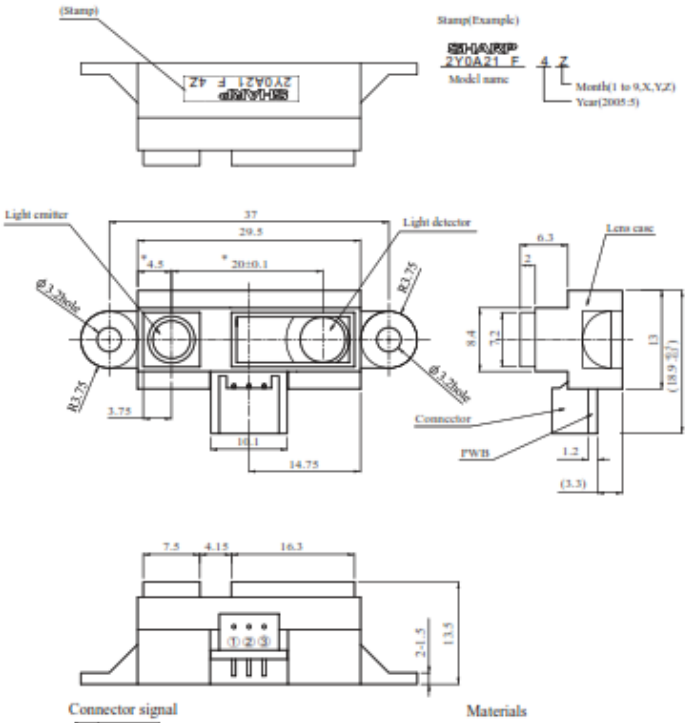
1. Touch-less switch
   (Sanitary equipment, Control of illumination, etc.)
2. Robot cleaner
3. Sensor for energy saving
   (ATM, Copier, Vending machine)
4. Amusement equipment
   (Robot, Arcade game machine)

■Block diagram



■Outline Dimensions

(Unit : mm)



Connector signal

	signal name
①	Vo
②	GND
③	Vcc

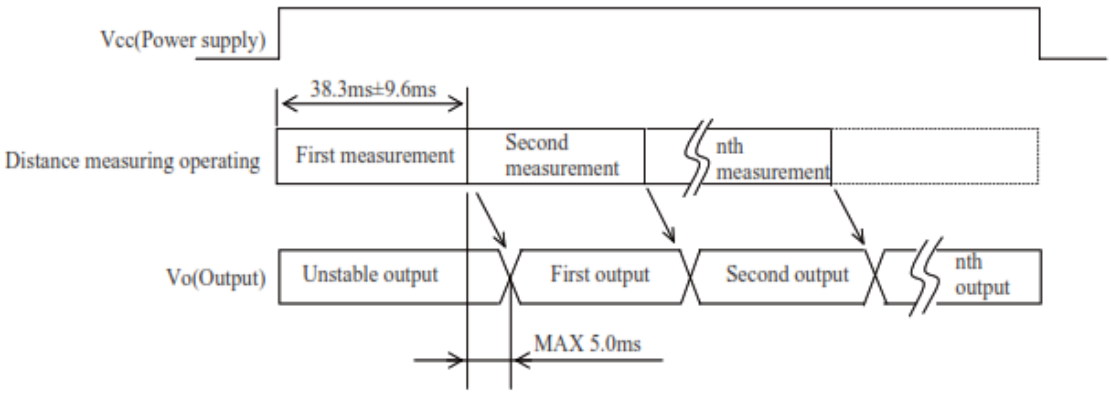
Connector :
J.S.T.TRADING COMPANY,LTD,
S3B-PH

Materials
Lens :Acrylic acid resin
(Visible light cut-off resin)
Case :Carbonic ABS
(Conductive resin)
PWB :Paper phenol

Note 1. The dimensions marked * are described the dimensions of lens center position.
Note 2. Unspecified tolerances shall be ± 0.3 mm.
Note 3. The dimensions in parenthesis are shown for reference.

Product mass : Approx. 3.6g

Fig. 1 Timing chart



■Absolute Maximum Ratings

(T_a=25℃,V_{CC}=5V)

Parameter	Symbol	Rating	Unit
Supply voltage	V_{CC}	-0.3 to +7	V
Output terminal voltage	V_O	-0.3 to V_{CC}+0.3	V
Operating temperature	T_{opr}	-10 to +60	℃
Storage temperature	T_{stg}	-40 to +70	℃

■Electro-optical Characteristics

(T_a=25℃,V_{CC}=5V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Average supply current	I_{CC}	L=80cm (Note 1)	—	30	40	mA
Distance measuring	ΔL	(Note 1)	10	—	80	cm
Output voltage	V_O	L=80cm (Note 1)	0.25	0.4	0.55	V
Output voltage differential	ΔV_O	Output voltage differece between L=10cm and L=80cm (Note 1)	1.65	1.9	2.15	V

* L : Distance to reflective object

Note 1 : Using reflective object : White paper (Made by Kodak Co., Ltd. gray cards R-27・white face, reflectance; 90%)

■Recommended operating conditions

Parameter	Symbol	Rating	Unit
Supply voltage	V_{CC}	4.5 to 5.5	V