Sistema de Control de Iluminación de Luces LED

Francisco Xavier Córdova De La Cruz

Universidad Internacional SEK

Nota de Autor

Francisco Xavier Córdova De La Cruz, Facultad de Ingeniería Mecánica, Universidad Internacional SEK; Director Ing. Gustavo Moreno. M.Sc.

Cualquier correspondencia concerniente a este trabajo puede dirigirse a: francisco041993@hotmail.com

Declaración Juramentada

Yo, FRANCISCO XAVIER CÓRDOVA DE LA CRUZ, con cédula de identidad 171816686-9, declaro bajo juramento que el trabajo aquí desarrollado es de mi autoría, que no ha sido previamente presentado para ningún grado a calificación profesional; y, que se ha consultado las referencias bibliográficas que se incluyen en este documento. A través de la presente declaración, cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la UNIVERSIDAD INTERNACIONAL SEK, según lo establecido por la Ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

FRANCISCO XAVIER CÓRDOVA DE LA CRUZ

C.I.: 171816686-9

Dedicatoria

Esta tesis se la dedico a:

Dios, por haberme permitido llegar hasta este punto y haberme brindado salud para cumplir mis objetivos.

A mis padres, que por su esfuerzo he llegado a culminar mis estudios Universitarios, siempre han sido un gran ejemplo y una guía fundamental en mi vida. Con su cariño y comprensión han logrado criarme como una persona de bien, responsable, honesto y perseverante.

A mi hermano, que ha sido un excelente ejemplo a seguir. Gracias a él siempre me he esforzado, alcanzando muchas metas propuestas. Sin duda, ha sido el apoyo más grande que Dios me pudo dar. Le pido a Dios que me conceda la dicha de siempre estar junto a mi familia, y que me permita recompensarles por todo lo que me han dado.

A todas las personas que han estado conmigo a lo largo de mi carrera.

Índice de contenidos

Índice de tablas y figuras

Tablas

Figuras

Resumen

Como resultado de este trabajo de investigación, se diseñó un sistema de control y monitoreo para luminarias utilizando radiofrecuencia mediante el estándar de comunicación IEEE 802.15.4. (zigbee). Este estándar fue elegido ya que permite realizar conexiones multipunto con un bajo ciclo de trabajo, obteniendo de esta manera un bajo consumo energético. Para esto se propuso la utilización de un microcontrolador Arduino UNO junto con 3 módulos Xbee conectados a dos luminarias. Para llevar a cabo la programación de los módulos y la interfaz de usuario, se utilizaron tres diferentes tipos de software: Arduino, Processing y Xctu. Durante la construcción, se realizaron dos experimentos que demostraron el correcto funcionamiento de la red Zigbee.

Palabras clave: Red mesh, Control de luminarias, X-ctu, Zigbee, Xbee S2, Arduino

Abstract

As a result of this research, a system that uses long-distance radio frequency was built to control and monitor luminaires by the standard communication IEEE 802.15.4. This standard was chosen because it allows multipoint connections with a low duty cycle, obtaining low energy consumption. For this project, the main devices used were: an Arduino UNO microcontroller and three Xbee modules. The Xbee modules were used to create the network of luminaries. To develop the backend software modules and the user interface we used the three following software suites: *Arduino*, *Processing* and *Xctu*. After construction, our experiments demonstrated the correct operation of the Zigbee network.

Introducción

Desde sus inicios, el alumbrado público ha sido un sistema vital, capaz de brindar luz durante toda la noche a los espacios abiertos como carreteras, parques, sitios públicos, etc. Actualmente, en la ciudad de Quito-Ecuador no existe ningún sistema de monitoreo de las luminarias, por tanto no es posible conocer si estas se encuentran en buen o mal estado. La falta de alumbrado público puede causar aumento en los accidentes de tránsito, en la delincuencia y afectar al turismo dependiendo de la ubicación. Como ejemplo, se puede citar publicaciones escritas por la prensa, como:

> La falta de alumbrado público preocupa a las personas que viven alrededor del Mirador de Guápulo, en la González Suárez, quienes denuncian que desde hace una semana los delincuentes están al acecho y aprovechan la oscuridad que hay en el sector (La Hora, 2012).
>
> El asesor técnico de seguridad vial de Aneta, Jesús Gómez, indica que el mismo hecho de conducir en la noche incrementa de tres a cuatro veces la posibilidad de que se presenten accidentes de tránsito. Esta probabilidad crece cuando las vías carecen de luminarias, afirma Gómez, debido al mal estado de la señalización en la ciudad. Indica que cuando no hay luz, los automovilistas tienden a encender las luces altas y no reducen su intensidad ante la llegada de otros automotores, por lo que limitan la visión de sus tripulantes. (El comercio, 2010)

A nivel mundial, varias empresas trabajan en soluciones para estos problemas, una de ellas es Circutor, SA. Esta empresa se encuentra desarrollando el sistema CirLAMP, que permite la gestión inteligente del alumbrado público a través de módulos manager con tecnología PLC (Power Line Communications), los cuales aprovechan la línea eléctrica existente de la instalación para comunicarse con módulos Nodo, instalados en cada punto de luz de la vía pública. (Circutor, p. 3).

El objetivo de este proyecto, es comprobar la factibilidad de realizar un sistema inteligente para monitorear las luminarias, enviando y recibiendo datos sin la necesidad de conexiones físicas, como los cables entre luminarias, lo cual a su vez ayuda con la tendencia

urbanística actual que centra su atención en el soterramiento de cables, ya que se reducen el impacto visual y aumentan la seguridad en las ciudades.

Dentro de los medios de transmisión de datos inalámbricos, se pueden mencionar tres tipos principales: la radiofrecuencia (RF), infrarrojos y laser, de los cuales el de radiofrecuencia es el más utilizado porque no necesita tener línea de vista directa para transmitir datos y se pueden crear grandes redes multipunto en comparación a los infrarrojos y laser. La RF utiliza ondas electromagnéticas que se propagan a través del espacio libre. Para su transmisión y recepción, será necesario que las antenas estén sintonizadas a la frecuencia de trabajo. Es de vital importancia conocer que la radiofrecuencia posee un alcance determinado, y por tanto, fuera del mismo se pierde la cobertura (Montero, 2014, p. 66)

Dentro de la radiofrecuencia, se pueden encontrar redes Zigbee, Bluetooth y Wi-fi. Las redes Zigbee tienen la capacidad de conectar una gran cantidad de nodos, aproximadamente 65000 distribuidos en subredes de 255 nodos, con relación a Bluetooth que solo permiten un máximo de 80 distribuidos en subredes de 8 nodos. Por otra parte, la topología de Zigbee va a permitir que se realicen redes con topología tipo Cluster Tree como se puede observar en la figura 1, mientras que Wifi permiten configuraciones tipo estrella que son una desventaja para la solución del problema analizado.

Por los motivos mencionados anteriormente, se decidió usar una red Zigbee, la cual se define como un protocolo de comunicación para las redes inalámbricas de corto alcance y baja velocidad de transmisión de datos. La velocidad de transmisión de datos es de 250K bit por segundo. Este protocolo va dirigido principalmente a las aplicaciones que requieran estar alimentadas por una batería, donde los principales requerimientos sean tener una transmisión baja de datos, tener un bajo costo y se necesite una larga duración de la batería. (Farahani, 2008, p. 1)

Nodos Zigbee

El protocolo de comunicación Zigbee, establece que pueden existir tres tipos de nodos, como puede observarse en la figura 1:

- Coordinadores: Son los nodos principales que establecen la conexión en la red y existe un solo coordinador por malla. Los coordinadores almacenarán la información de la red, incluyendo los datos de seguridad. (Digi, 2014)
- Routers: Los routers son dispositivos intermedios utilizados principalmente para extender la red. (Digi, 2014)
- Dispositivos finales: Los dispositivos finales, usualmente están conectados a sensores que pueden ser de bajo consumo o estar alimentados por una fuente externa, como una batería. Estos nodos, pueden comunicarse con routers y coordinador, pero no con otros dispositivos de mismo rango. (Digi, 2014)

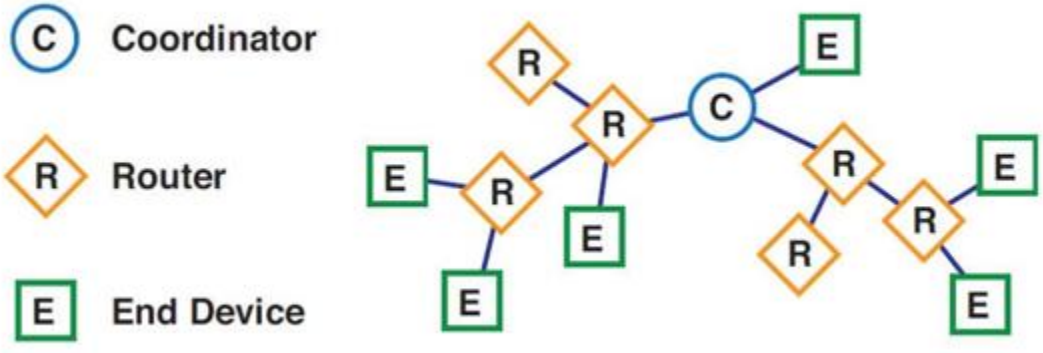

Figura 1. Ejemplo de una red Zigbee (Cluster Tree) obtenida de (Digi, 2014)

Configuración de Zigbee

Los módulos Xbee Zigbee pueden operar de dos maneras:

- AT: El modo AT, o modo transparente está limitado a la transmisión de datos "punto a punto" entre dos módulos Xbee. (Digi, 2014)
- API: El modo API, se encuentra diseñado para permitir el envío y la recepción de datos entre el coordinador y varios módulos Xbee. En cada trama enviada por los Xbee, se obtendrá una mayor variedad de información adicional codificada. (Digi, 2014)

Adicionalmente, se analizó la posibilidad de realizar un control de iluminación con la técnica Pulse-Width Modulation (PWM), para lo cual se consideró el uso del Xbee S2, sin embargo según se puede apreciar en la Tabla 1, los Xbee S2 no poseen salidas análogas (PWM) a diferencia de los Xbee S1, pero los Xbee S1 no permiten realizar redes tipo malla lo cual es fundamental para solucionar el problema planteado.

Xbee

Los módulos Xbee S2 de la compañía DIGI, poseen ventajas como las de tener incorporados pines digitales y análogos para enviar y recibir datos sin la necesidad de utilizar un microcontrolador en cada nodo. Los módulos pueden crear redes complejas multipunto además de poseer un bajo consumo de energía y de permitir una comunicación simple entre el computador y el Xbee, facilitando de esta manera su programación y actualización de firmware. "Los módulos poseen un alcance de 40 metros en la zona urbana" (MaxStream, 2007, p. 5), lo cual cumple con la ordenanza municipal de Quito, que cita: "los postes deben estar ubicados a una separación de 30 m entre sí" (Comisión de planificación y nomenclatura, 31, p. 67).

Características especiales entre Xbee Serie 1 (802.15.4) y Xbee Serie 2 (Zigbee)

Tabla 1

Diferencia entre Xbee Serie 1 y Xbee Serie 2

XBEE SERIE 1	XBEE SERIE 2
8 Digital i/os	10 Digital I/Os
7 Entradas análogas	4 Entradas análogas
2 Salidas análogas (PWM)	No tiene salidas análogas (PWM)
Se debe conectar Vref hasta 3.3V	No se debe conectar Vref, entrada de voltaje hasta 1.2V
Topología de red punto a punto	Topología de red permite conexiones Cluster Tree

Fuente: (Shorter, p. 22)

Sensores de Corriente

Dentro del mercado actual, se pueden encontrar una variedad de sensores de corriente de efecto Hall invasivos como el ASC712, que se los puede encontrar en rangos de hasta 5 A, 20 A, 30 A y sensores de efecto Hall no invasivos como Sct 013-000 de hasta 100 A.

En este proyecto, se escogió el sensor de corriente ASC712 por su reducido costo, facilidad de compra en el mercado y porque no se tendrán corrientes altas en las luminarias.

El sensor de corriente, se utilizará para conocer si una luminaria se encuentra en mal estado midiendo la corriente que pasa por la misma. El sensor ASC712, “es un sensor de

corriente que provee una solución económica y precisa para realizar trabajos que necesiten una fácil implementación como control de motores, detección y administración de carga, protección ante el exceso de corriente, etc." (Allegro MicroSystems). Así, cuando no exista una señal en dicho sensor se podrá concluir que existe una avería dentro del nodo.

Arduino

En este proyecto se utilizó la plataforma y la placa de Arduino por su facilidad de programación y facilidad de realizar conexiones eléctricas, considerando además que el costo del micro controlador Arduino UNO es de los más económicos en el mercado. Arduino es una plataforma que permite la creación de códigos libres, con un software y hardware fácil de utilizar. Se puede decir que el usuario le puede decir al micro controlador que hacer, solamente utilizando el lenguaje de programación de Arduino (basado en Wiring) y el software de Arduino IDE (Integrated Development Environment), basado en Processing. (Arduino, s.f.)

Processing

En este proyecto, se implementó una interfaz gráfica para mejorar la interpretación de datos que van a ser entregados al usuario y se eligió a Processing, ya que permite una fácil comunicación con Arduino. "Processing es un software libre basado en Java, diseñado para generar artes visuales, animaciones y cualquier tipo de aplicaciones gráficas" (Linares, s.f.)

X-ctu

En este proyecto, se utilizó la plataforma X-ctu para realizar todas las configuraciones en los módulos Xbee S2, actualizar Firmware e interactuar fácilmente con los módulos.

"X-ctu es una plataforma libre diseñada para permitir a los desarrolladores interactuar con los módulos Digi RF a través de una simple interfaz gráfica. Incluye también herramientas que facilitan la preparación y configuración de los módulos Xbee RF. " (Digi, s.f.)

Método

Diseño

Se creó una red Zigbee (cluster tree), mediante la cual se puede controlar y monitorear las luminarias a través de un software desarrollado en Arduino y Processing. Para lograrlo, se ubicó un módulo Xbee S2 en cada luminaria, al cual están conectados: un sensor de corriente, un transistor y un relé para controlar el encendido y apagado de la luminaria. Finalmente, un Xbee S2 está conectado a un microcontrolador Arduino UNO que será el encargado de coordinar la red mesh mediante la interfaz creada en el software Processing.

El Xbee coordinador, irá conectado a un microcontrolador Arduino UNO y este a su vez, a un computador. De esta manera, a través del programa creado en Processing (Anexo B), se podrá monitorear el estado de las luminarias así como, encenderlas y apagarlas de una manera eficiente.

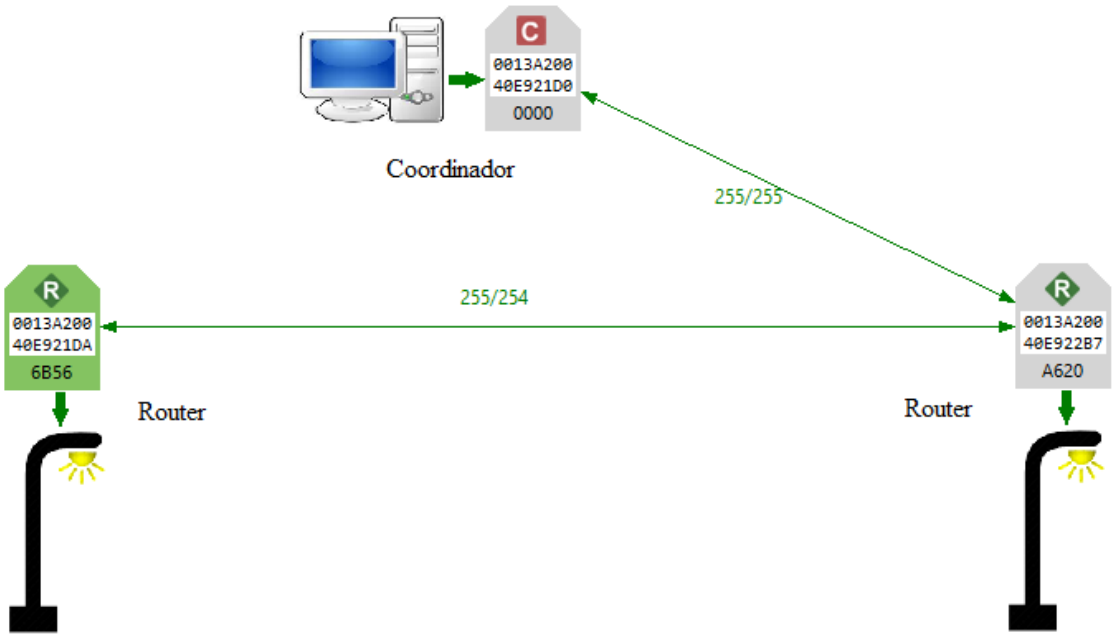

Figura 2. Esquema general del sistema

Software

Se utilizaron las plataformas Arduino (versión 1.6.3) y Processing (versión 2.2.1) para desarrollar la programación que va a estar dividida en: lectura y envío de datos. El código de programación de Arduino (Anexo A), va a coordinar la recepción de datos de todos los Xbee existentes en la red multipunto y del envío de órdenes, tales como encender o apagar las luminarias. Por otro lado, será necesario incorporar una pantalla que permita la comunicación amigable entre máquina-usuario que fue desarrollada en Processing.

Envío de datos

En la figura 3, se puede observar el esquema de cómo será el proceso de envío de datos.

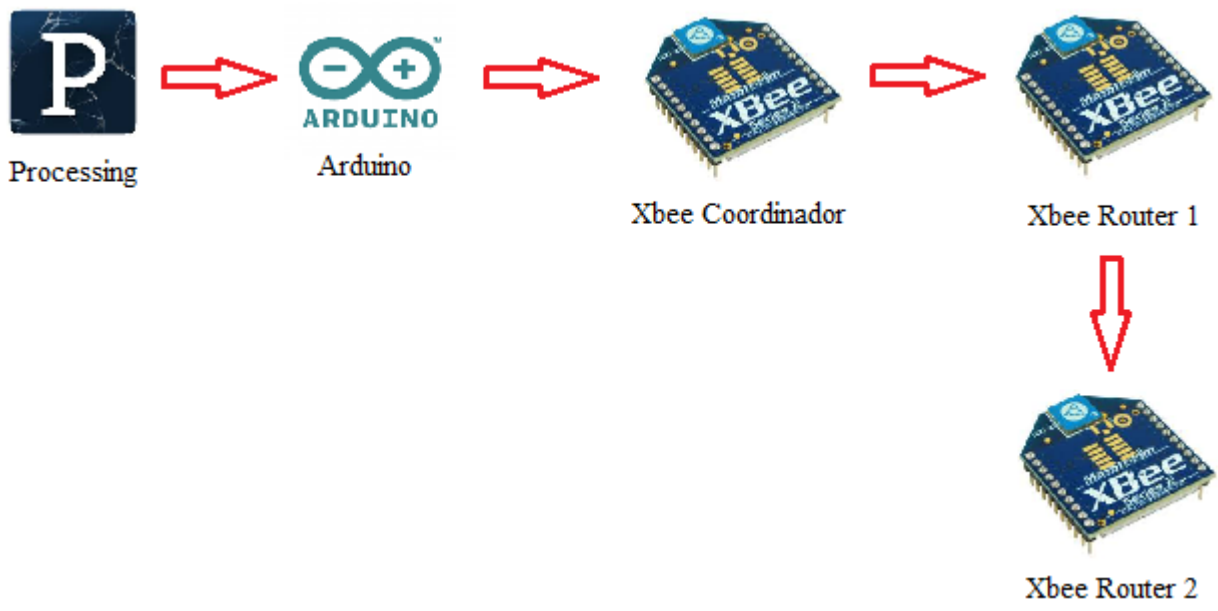

Figura 3. Esquema general del envío de datos

Envío de datos de Processing a Arduino

Se desarrolló una interfaz gráfica en Processing, en la cual se podrá visualizar la cantidad de corriente que está circulando por cada luminaria, además que se dispondrá de 4 botones, dos de encendido y dos de apagado para que el usuario tenga una mayor facilidad de controlar las

luminarias. Como se puede apreciar en la figura 4, los parámetros para que se activen los botones serán que el mouse se encuentre dentro del cuadrado y que el usuario de un clic en el mismo. Las órdenes se enviarán a través del puerto serial hacia Arduino y serán números del 1 al 4 seguidos de un punto, dependiendo del botón.

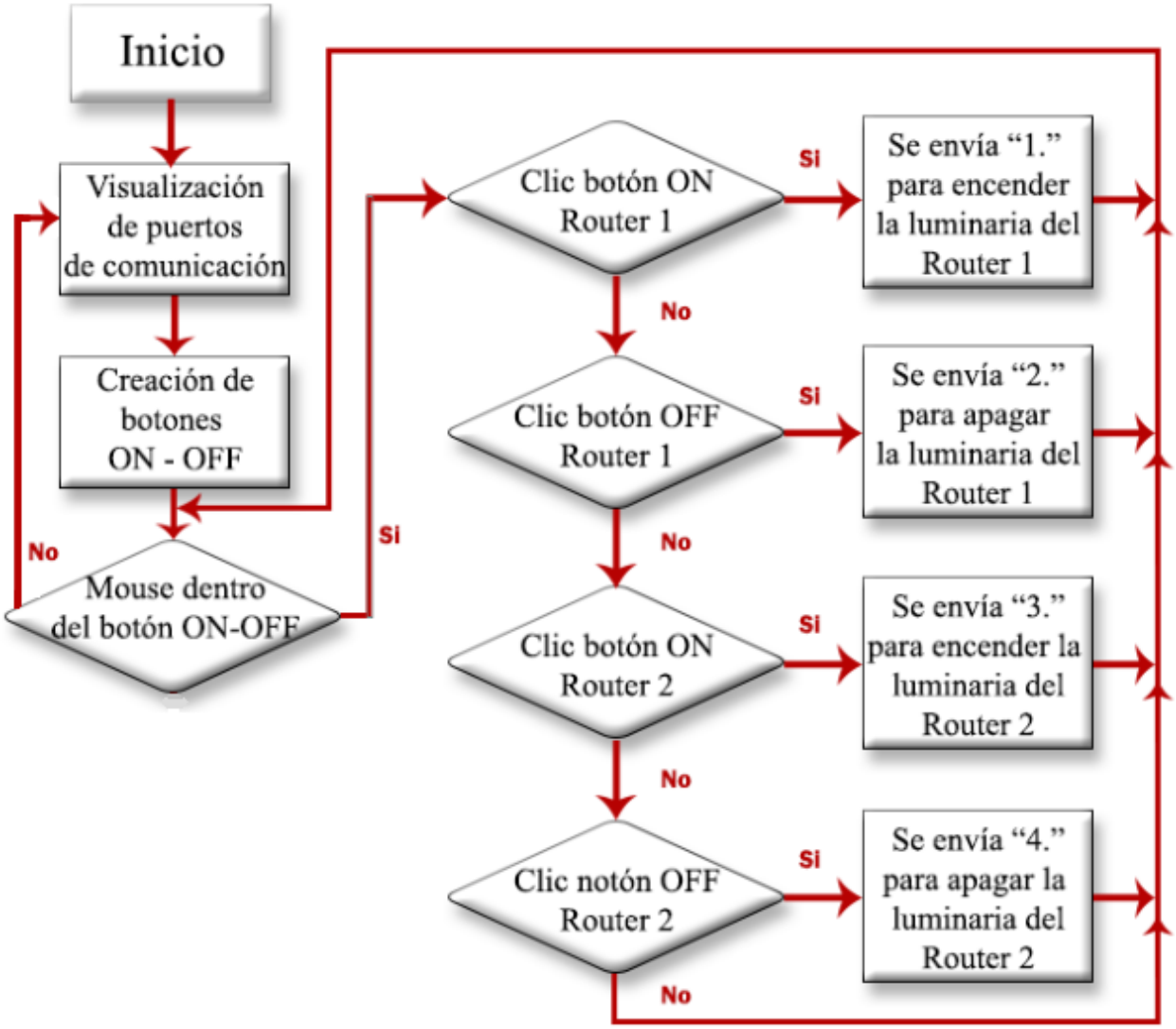

Figura 4. Algoritmo para encender o apagar luminarias en Processing

Envío de datos de Arduino al Xbee Coordinador y a los Xbee Router

Para encender o apagar las luminarias, se creó un código en Arduino (Anexo A) en el cuál se cambia el estado del pin Digital D0 de High a Low y viceversa. Para esto, se debió utilizar el formato API (Tabla 2).

Parámetros que se modifican en la trama de envío de datos:

- El primer parámetro que se va a modificar será la dirección MAC (Media Access Control) de los módulos, que serán los bytes del 5 al 12 para conocer a que Xbee Router se enviará la trama.
- Dentro del comando AT (byte 16 y 17) se nombra al pin digital D0, ya que este fue el que se activó para enviar la señal de encender o apagar las luminarias
- En el byte 18, pueden existir dos opciones: 0x05 para encender la luminaria y 0x04 para apagar la luminaria
- El byte 19 es la suma de comprobación, esta de igual manera va a variar conforme se modifiquen los bytes anteriores.

Tabla 2

Formato de Interfaz de Programacion de Aplicación (API)

Byte	**Ejemplo**	**Descripción**
0	0x7E	Delimitador
1 **2**	0x00 0x10	Tamaño de la muestra
3	0x17	Identificación de la muestra, 0x17 significa que es una solicitud
4	0x00	Identificación de la muestra
5 **6** **7**	0x00 0x13 0xA2	Dirección del Xbee que envía la muestra

8 **9** **10** **11** **12**	0x00 0x40 0xE9 0x21 0xD7	
13 **14**	0xFF 0xFE	Dirección del Xbee que envía la muestra
15	0x02	Opciones de la solicitud (Se aplica 0x02 para aplicar los cambios
16 **17**	**‘D’** **‘0’**	Nombre del comando AT
18	**0x05**	Parámetro del comando
19	**Checksum1= (0x17 + 0x00 + 0x13 + 0xA2 + 0x00 + 0x40 + 0xE9 + 0x21 + 0xD7 +0xFF + 0xFE + 0x02 + 'D' + '0' + 0x05);** **Checksum (0xFF - (checksum1 & 0xFF));**	Suma de comprobación

Extraído de (Digi) y modificado con datos experimentales

Como se puede observar en la figura 5, los datos que se receptarán de Processing a través del puerto serial en Arduino, servirán para conocer a que luminaria se la va a encender o a apagar. Los datos (1.) y (3.) encenderán las luminarias mientras que (2.) y (4.) apagarán las luminarias respectivamente.

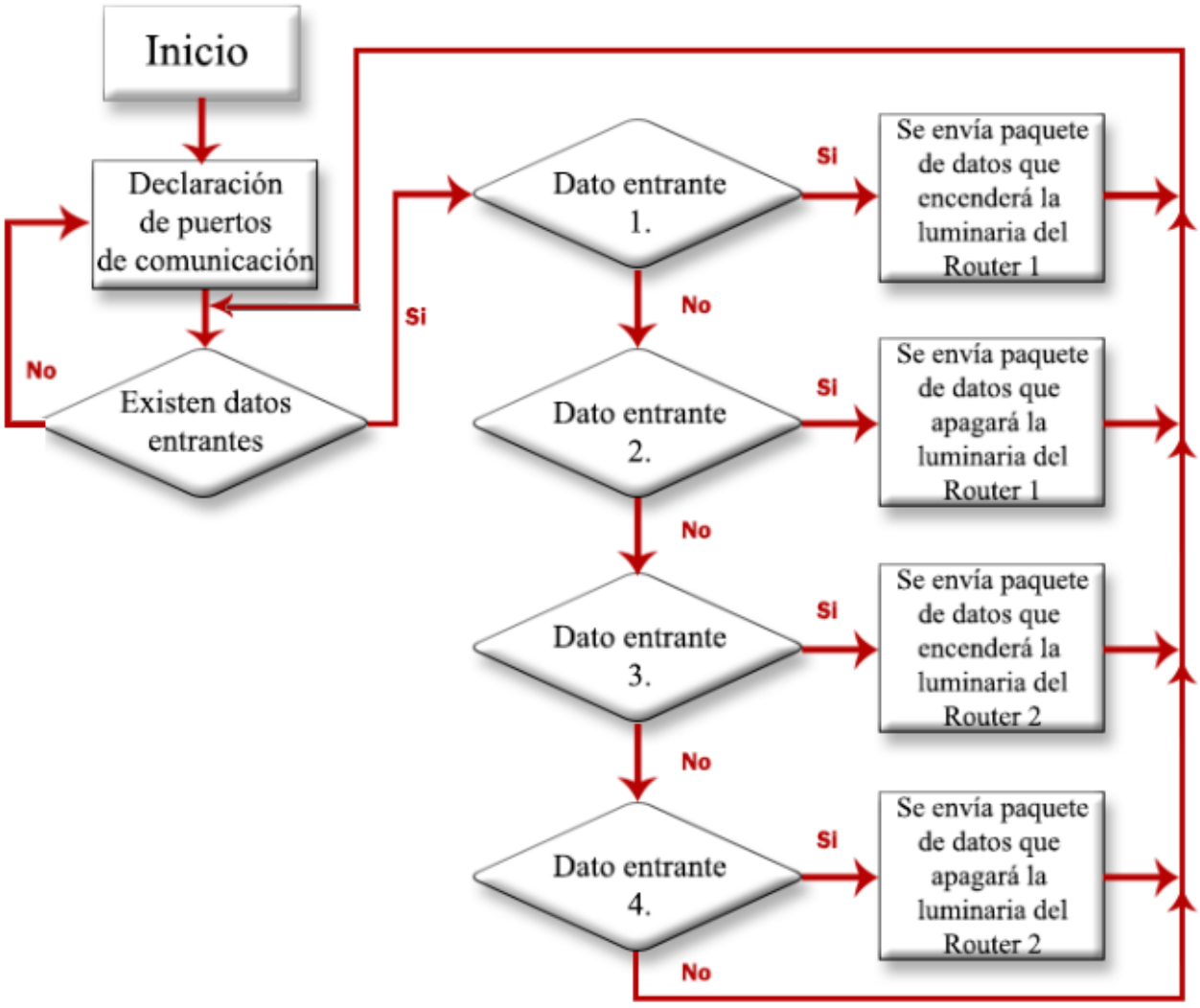

Figura 5. Algoritmo para encender o apagar las luminarias en Arduino

Recepción de datos

En la figura 6, se puede observar el esquema del proceso de recepción de datos.

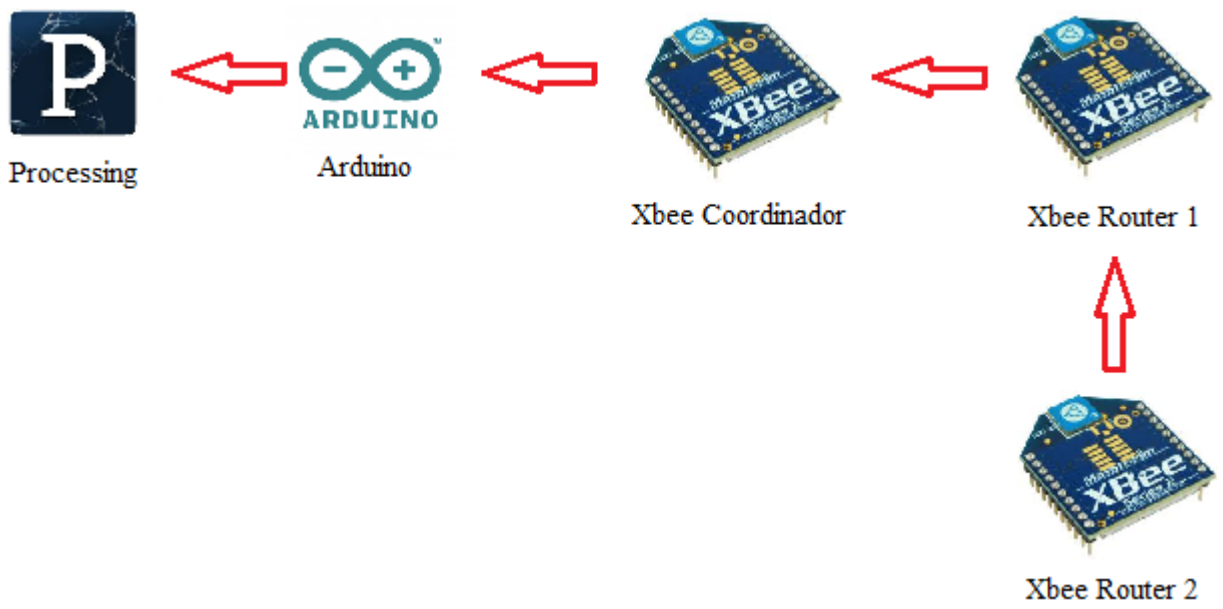

Figura 6. Esquema general de recepción de datos

Los datos serán enviados desde el Router 1 o el Router 2 hacia el Xbee Coordinador y luego al Arduino. Para la lectura de datos de los sensores de corriente, será necesario identificar la dirección de donde provienen y poder distinguir que dato pertenece a cada luminaria. Para filtrar esta información, se toma como referencia la dirección MAC (Media Access Control) de 64 bits que traen los Xbee de fábrica, la cual es única.

Dentro del diagrama que se puede encontrar en la figura 7, primero se verifica que exista un paquete de datos entrante al Xbee coordinador, para luego filtrar el delimitador 0x7E como se muestra en la tabla 3, Ítem 1, donde se comenzará a analizar la información recibida. Como se puede observar en los datos de (64-bit, dirección de fuente) del Router 1 (Tabla 3, Ítem 2) y Router 2 (Tabla 3, Ítem 3), luego del delimitador 0x7E, los siguientes 9 bytes son idénticos hasta llegar al décimo, en el cual se podrá distinguir como Router 1 al byte 0x21 y Router 2 al byte 0x22. Posterior a esto, se eliminan los siguientes 10 bytes hasta encontrar la muestra análoga, que serán los dos siguientes bytes.

Tabla 3

Explicación de la trama de datos entrantes al Xbee Coordinador

Ítem	**Nombre del campo**	**Muestra (HEX)**	**Descripción**
1	Delimitador	0x7E	Todas las muestras poseen el mismo delimitador para conocer donde empieza cada paquete de datos recibido
2	64-bit, dirección de fuente Xbee 1	0x00 0x13 0xA2 0x00 0x40 0xE9 **0x21** 0xD7	Dirección del Xbee que envía la muestra
3	64-bit, dirección de fuente Xbee 2	0x00 0x13 0xA2 0x00 0x40 0xE9 **0x22** 0xB7	Dirección del Xbee que envía la muestra

Tabla extraída de resultados del experimento 1

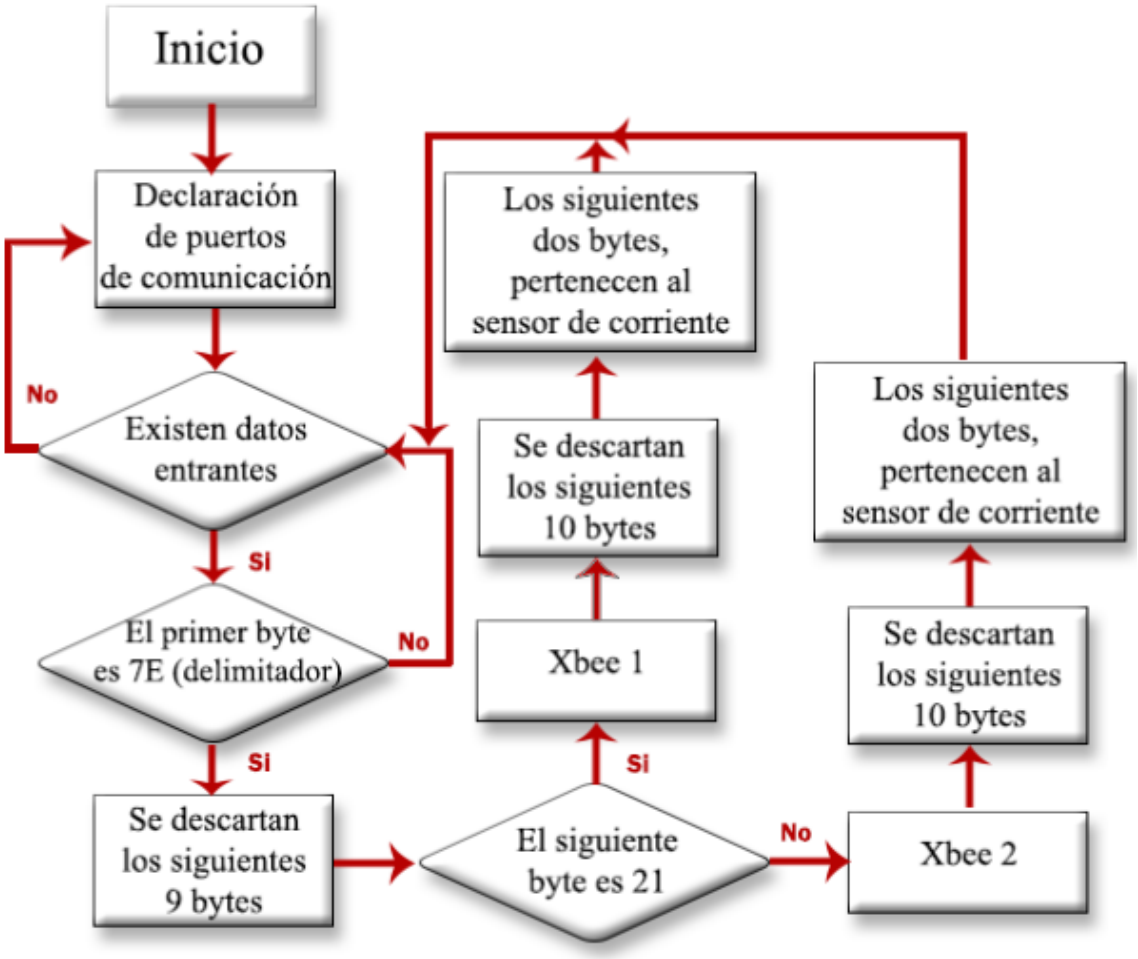

Figura 7. Algoritmo de lectura de datos en Arduino

Envío de datos de Arduino a Processing.

Como punto final para la recepción de datos, se puede observar en la figura 8, que la información receptada en la plataforma Arduino, se envía a Processing a través del puerto serie utilizando delimitadores antes del valor análogo para diferenciar las luminarias. Si el delimitador antepuesto al valor del sensor de corriente es 0x7E, este corresponderá al Router 1, si el valor es 0x7F, corresponderá al Router 2. En esta plataforma, el usuario podrá visualizar de mejor manera la cantidad de corriente que está circulando en las luminarias.

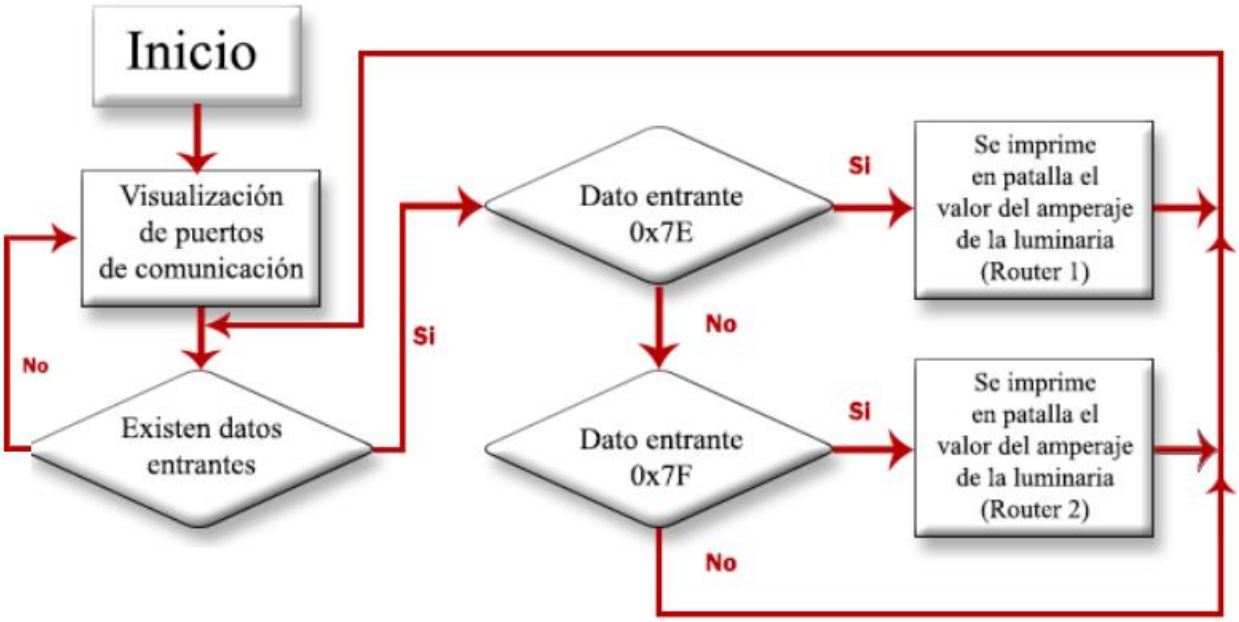

Figura 8. Algoritmo de lectura y visualización de datos en Processing

Hardware

Coordinador

En la figura 9, se puede apreciar el esquema de conexión del módulo Xbee S2 (Coordinador) con el microcontrolador Arduino UNO, que será el vínculo entre el computador y los módulos Xbee. La conexión es a través de los pines TX y RX de ambos módulos.

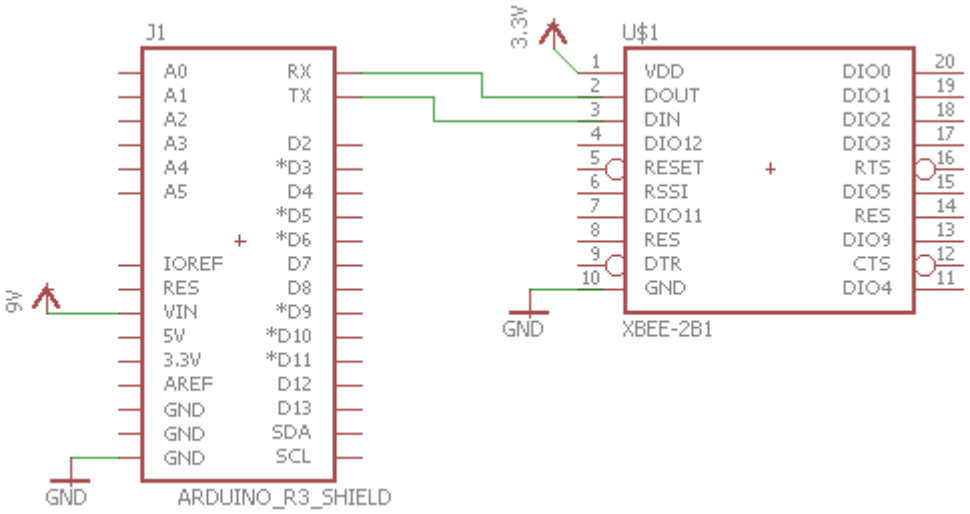

Figura 9. Conexión Xbee Coordinador.

Router

En la figura 10 se puede observar el diagrama de conexión entre la luminaria y el Xbee S2. Dentro del esquema, se puede apreciar que el módulo Xbee y el sensor de corriente ACS712 están conectados a través del pin DI02.

Por otro lado, debido al bajo voltaje y corriente que puede proveer el chip Xbee, se utilizó un transistor TIP31 como etapa de media potencia para encender el relé de activación de la luminaria LED. El circuito utilizado se puede observar en la figura 10.

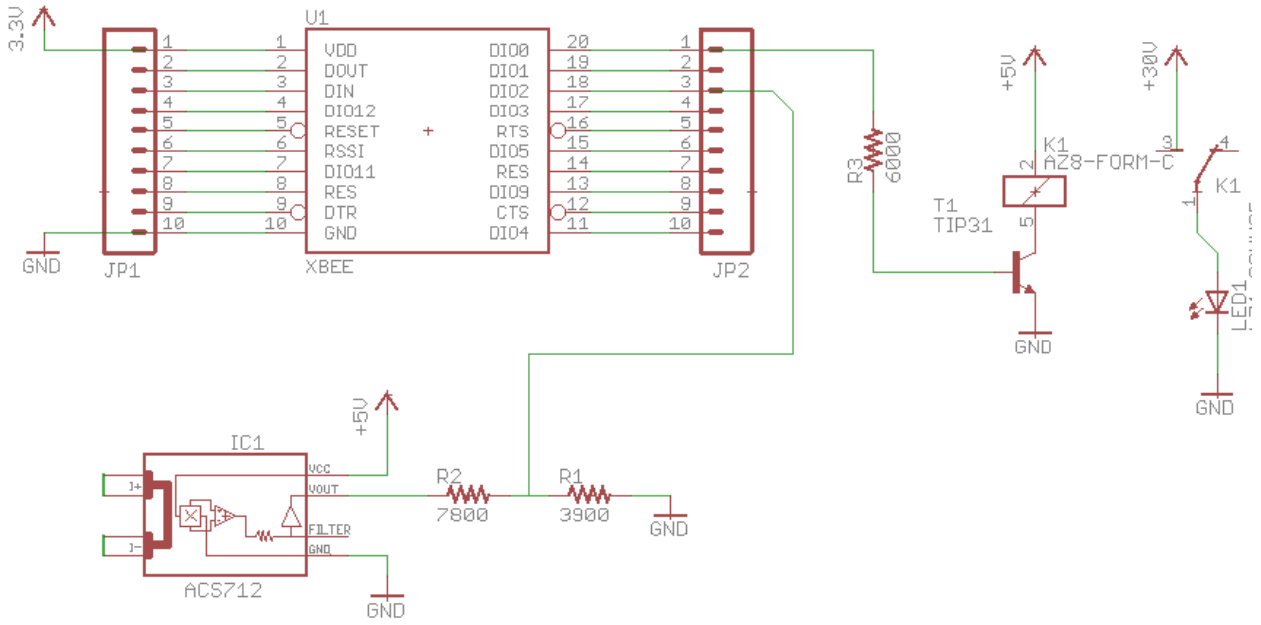

Figura 10. Conexión Xbee Router 1 y 2

Cálculo de transistor que accionará el relé

Se utilizó un transistor NPN TIP31C como se observa en la figura 10 para controlar la bobina del relé. Los cálculos del circuito se muestran en las ecuaciones (1) y (2):

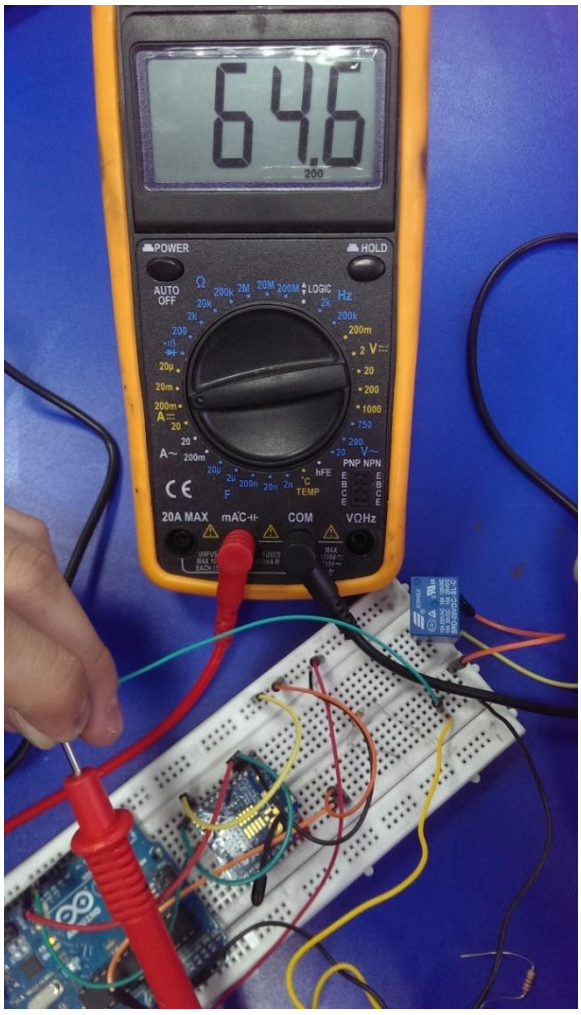

Figura 11. Medición experimental de la intensidad necesaria para el accionamiento de la bobina del relé

Ecuación aproximada para el cálculo de la resistencia que se utilizará en la base del colector

La corriente necesaria en el colector para activar el relé se la obtuvo experimentalmente como se muestra en la figura 11, con un valor de 0,065 A

$$Rb = \frac{(Vi - 0{,}7) * hFe}{Ic} \quad (1)$$

Dónde:

Ic = Corriente en el colector (A)

Rb = Resistencia de la base (Ω)

Vi = Voltaje de la salida digital del Xbee S2 (V)

hFe = Ganancia de corriente del transistor

hFe = 150 (Fairchild, 2014)

$$Rb = \frac{(3.3 - 0{,}7) * 150}{0{,}065} = 6000\Omega \quad (2)$$

Cálculo de la corriente base

Se calculó la corriente que va a utilizar el transistor en su base como se puede observar en las ecuaciones (3) y (4), para comprobar que es menor a la corriente que podrá entregar el módulo Xbee, la cual es de máximo 2mA.

$$Ib = \frac{Ic}{hFe} \quad (3)$$

$$Ib = \frac{0.065}{150} = 0.43mA \quad (4)$$

Cálculo de divisor de voltaje en el sensor de corriente ASC712

Los Xbee S2 poseen entradas análogas que soportan voltajes hasta 1.2 V, sin embargo se utilizará en sensor de corriente ASC712 con una máxima medida de amperaje de 5A que, como se aprecia en la figura 12, entrega un voltaje de salida de 3.4 V máximo. Como el voltaje excede al permitido por el módulo Xbee, se diseñó un circuito divisor de voltaje para cambiar los rangos de 0 - 3.4V a 0 - 1.2V.

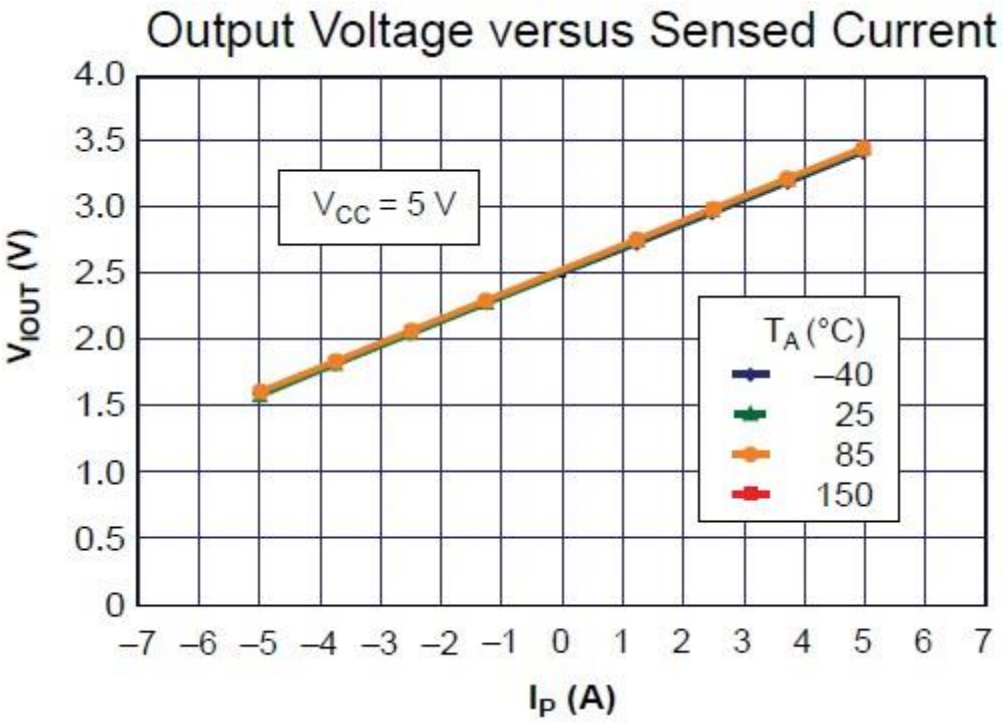

Figura 12. Relación entre voltaje de salida y corriente detectada obtenida de (Sparkfun, 2006)

Ecuación para calculo del divisor de voltaje

$$Vsalida = \left(\frac{R1}{R1 + R2}\right) * Vcc \quad (3)$$

Dónde:

Vsalida = El voltaje a obtener deberá ser de 1.2V.

R1 = Resistencia 1

R2 = Resistencia 2

Vcc = Voltaje de entrada (Voltaje máximo que puede entregar el sensor de corriente)

$$Vsalida = \left(\frac{3900}{3900 + 7800}\right) * 3.4 \quad (4)$$

Como resultado, para obtener un Vsalida de 1.2 V se deben utilizar resistencias que satisfagan la ecuación 3 para lo cual se seleccionaron resistencias de:

R1 = 3900 (Ω)

R2 = 7800 (Ω)

Experimento 1

El objetivo de este experimento, es comprobar si las dos únicas variables que se modifican conforme se activen o no los puertos digitales (D0, D2), son los bytes de muestras digitales y la suma de comprobación.

Método y Ejecución del Experimento

Variables y parámetros.

Parámetros:

- Se tomarán muestras en intervalos de 1 segundo.
- Potenciómetros en 0V

Las variables a utilizar son:

- D0 “High”.
- D0 “Low”.

Equipo y Materiales.

- 3 Xbee S2
- 2 Breakout board para Xbee
- 1 Xbee Explorer USB
- 1 Microcontrolador arduino UNO
- 2 Protoboards
- 2 Potenciómetros de 10k
- 1 Multímetro

Desarrollo del experimento.

Como primer paso, se debe configurar los Xbee de la siguiente manera:

- Un Xbee como coordinador
- Dos Xbee como router.

Para configurar cualquier Xbee, es necesario disponer de un Xbee Explorer, ya que este va a permitir la comunicación del Xbee con el software X-Ctu y seguir los siguientes pasos:

1. Se debe instalar el software X-Ctu en el computador.
2. Colocar el Xbee S2 en el Xbee Explorer, y conectarlo en la entrada USB del computador.
3. Abrir el programa X-Ctu y seleccionar la opción "Discover radio modules connected to your machine".
4. Seleccionar el puerto en el cual fue conectado el Xbee.
5. La configuración predeterminada de los Xbee, es de 9600 Baud rate. Se seleccionarán los parámetros de búsqueda y hacer clic en siguiente.
6. Una vez que el dispositivo sea reconocido, se lo deberá seleccionar y hacer clic en "Add selected devices".
7. Una vez que se lo haya agregado, se debe proceder a leerlo seleccionando la opción "Read"
8. Posterior a esto, se deben configurar los módulos Xbee con los parámetros de funcionamiento, dependiendo de su rango en la red (Coordinador-Router).

Coordinador:

- ID PAN ID: 3726247

Router 1

- ID PAN ID: 3726247
- JV Channel Verification: Enabled [1]
- D0 AD0/DIO0 Configuration: Variable
- D2 AD2/DIO2 Configuration: ADC [2]
- IR IO Sampling Rate: (3E8 X 1ms) – 1s

Router 2

- ID PAN ID: 3726247
- JV Channel Verification: Enabled [1]
- D0 AD0/DIO0 Configuration: Variable
- D2 AD2/DIO2 Configuration: ADC [2]
- IR IO Sampling Rate: (3E8 X 1ms) – 1s

Posterior a esto, se procede a conectar el Xbee coordinador al microcontrolador Arduino UNO para receptar y analizar los paquetes de datos entrantes como se puede observar en la figura 9. Los Xbee routers, se deben conectar a los Breakout board en dos protoboards distintos y los potenciómetros a las entradas en el pin D2 en el Router 1 y el Router 2, para simular el funcionamiento de los sensores ASC712 como se puede observar en la figura 10.

Experimento 2

El objetivo de este experimento, es comprobar la topología tipo Cluster Tree, enviando datos desde el Xbee coordinador al Xbee router 2, pasando por el Xbee router 1.

Método y Ejecución del Experimento

Variables y parámetros.

Parámetros:

- Se tomarán muestras en intervalos de 1 segundo.
- Se enviarán paquetes de datos que enciendan y apaguen las dos luminarias.

Las variables a utilizar son:

- Los tres dispositivos conectados, uno funcionando como coordinador y los otros dos como router.
- Desconectar el Xbee router 1 que se encuentra en la mitad de los dos dispositivos.

Equipo y materiales.

- 3 Xbee S2
- 2 Breakout board para Xbee
- 1 Xbee Explorer USB
- 1 Microcontrolador arduino UNO
- 2 Protoboards
- 2 Potenciómetros de 10k
- 1 Multímetro
- 2 LEDs

- 2 Resistencias de 330 ohm

Desarrollo del experimento.

Se ubicó a los 3 módulos Xbee según se observa en la Figura 13. Posterior a esto, desde el Xbee coordinador, se envió la señal para encender la luminaria del Xbee Router 1 y la luminaria del Xbee Router 2. Una vez comprobado que se encendieron las dos luminarias, se procede verificar que el módulo central funciona como router. Para esto se desconectó el Xbee Router 1 (módulo central) y nuevamente se envió la señal de encender la luminaria del Xbee Router 2, como resultado, esta no se encendió.

Figura 13. Ubicación de módulos Xbee en la Universidad Internacional SEK para experimento 2

Experimento 3

El objetivo de este experimento, es medir el consumo energético de los módulos Xbee conforme envíen y reciban datos para observar su variación y determinar el consumo energético que tendrá el circuito de control.

Método y Ejecución del Experimento

Variables y parámetros.

Parámetros:

- Se enviarán paquetes de datos que enciendan y apaguen las dos luminarias.
- Siempre se estará recibiendo datos de corriente de los Xbee Router

Las variables a utilizar son:

- Los tres dispositivos conectados, uno funcionando como coordinador y los otros dos como router.

Equipo y materiales.

- 3 Xbee S2
- 2 Breakout board para Xbee
- 1 Xbee Explorer USB
- 1 Microcontrolador arduino UNO
- 2 Protoboards
- 2 Potenciómetros de 10k
- 1 Multímetro
- 2 LEDs

- 2 Resistencias de 330 ohm

Desarrollo del experimento.

Con los 3 Xbee conectados en red, se procede a medir el amperaje que pasa por los módulos conectando un multímetro en serie (Figura 14). Posterior a esto, con el multímetro conectado, se procede a enviar tramas de datos a través de la interfaz creada en Processing para encender y apagar las luminarias observando los picos de corriente.

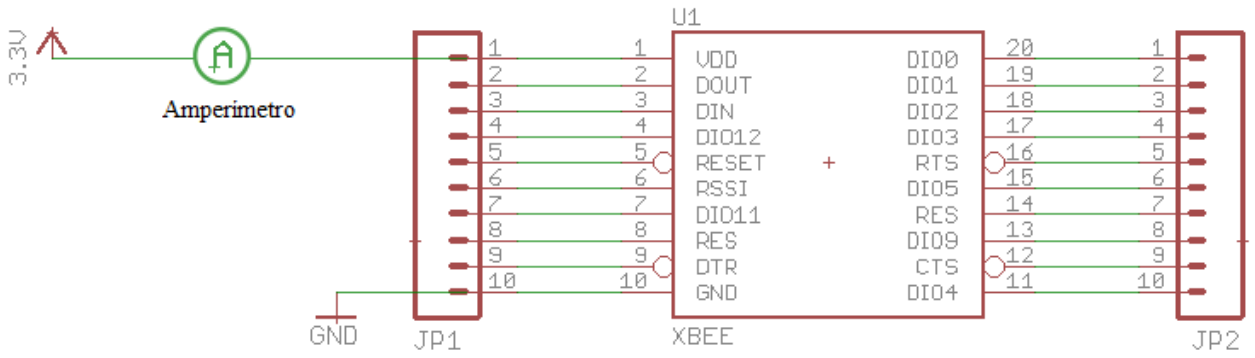

Figura 14. Circuito para medición de amperaje Xbee

Resultados

Experimento 1

Muestra de datos

Router 1, API (0013A20040E921D7)

Muestras de las lecturas análogas con distintos estados en la opción D0

- D0 “Low”.

  7E, 0, 14, 92, 0, 13, A2, 0, 40, E9, 21, D7, F7, FF, 1, 1, 0, 1, 4, 0, 0, 0, 0, 9A.

- D0 “High”.

  7E, 0, 14, 92, 0, 13, A2, 0, 40, E9, 21, D7, F7, FF, 1, 1, 0, 1, 4, 0, 1, 0, 0, 99.

Router 2, API (0013A20040E922B7)

Muestras de las lecturas análogas con distintos estados en el la opción D0

- D0 “Low”.

  7E, 0, 14, 92, 0, 13, A2, 0, 40, E9, 22, B7, 23, 57, 1, 1, 0, 1, 8, 0, 0, 0, 0, 31.

- D0 “High”.

  7E, 0, 14, 92, 0, 13, A2, 0, 40, E9, 22, B7, 23, 57, 1, 1, 0, 1, 8, 0, 1, 0, 0, 30.

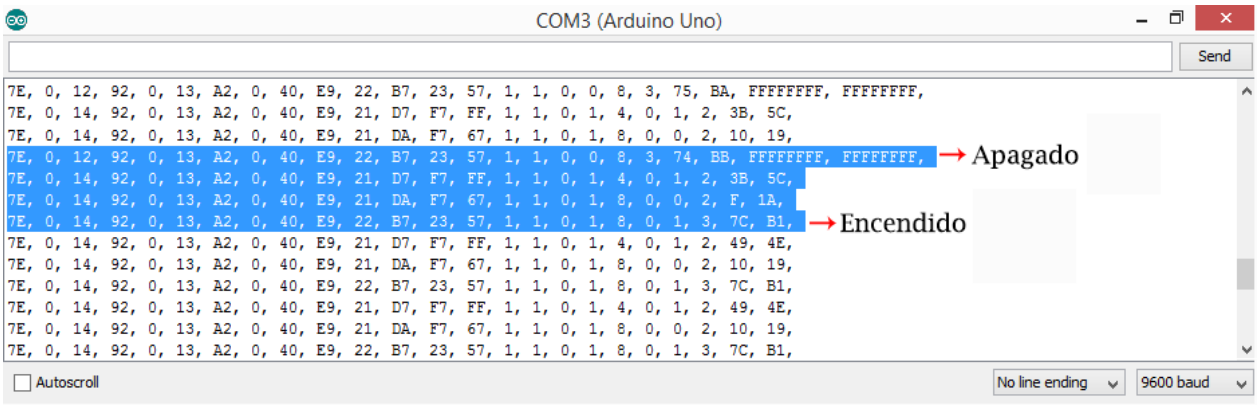

Figura 15. Datos recibidos en monitor serial de Arduino

Router 1

Tabla 4

Explicación de la muestra del router 1 con parámetro D0 "Low"

Nombre del campo	**Muestra (HEX)**	**Descripción**
Delimitador	0x7E	Todas las muestras poseen el mismo delimitador para conocer donde empieza cada paquete de datos recibido
Tamaño de la muestra	0x00 0x14	Tamaño de la muestra
Tipo de muestra	0x92	Identifica que la muestra fue automáticamente generada
64-bit, dirección de fuente	0x00 0x13 0xA2 0x00 0x40 0xE9 0x21 0xD7	Dirección del Xbee que envía la muestra
16-bit, dirección de fuente	0xF7 0xFF	Dirección del Xbee que envía la muestra
Opciones de recepción	0x01	Opciones de recepción
Número de muestras	0x01	Indica cuántas muestras fueron incluidas en el paquete
Digital channel mask	0x00 0x01	Indica los campos que fueron habilitados como DIO
Analog channel mask	0x04	Indica los campos que fueron habilitados como ADC

Muestras digitales	0x00 0x00	Contiene todas las muestras digitales
Muestras análogas	0x00 0x00	Existen dos bytes para cada pin seleccionado como ADC
Suma de comprobación	0x9A	Es la suma de toda la muestra API recibida

Extraído de (Digi) y modificado con datos obtenidos en experimento 1

Tabla 5

Explicación de la muestra del router 1 con parámetro D0 "High"

Nombre del campo	**Muestra (HEX)**	**Descripción**
Delimitador	0x7E	Todas las muestras poseen el mismo delimitador para conocer donde empieza cada paquete de datos recibido
Tamaño de la muestra	0x00 0x14	Tamaño de la muestra
Tipo de muestra	0x92	Identifica que la muestra fue automáticamente generada
64-bit, dirección de fuente	0x00 0x13 0xA2 0x00 0x40 0xE9 0x21 0xD7	Dirección del Xbee que envía la muestra
16-bit, dirección de fuente	0xF7 0xFF	Dirección del Xbee que envía la muestra
Opciones de recepción	0x01	Opciones de recepción
Número de muestras	0x01	Indica cuántas muestras fueron incluidas en el paquete
Digital channel mask	0x00 0x01	Indica los campos que fueron habilitados como DIO
Analog channel mask	0x04	Indica los campos que fueron habilitados como ADC
Muestras digitales	0x00 0x01	Contiene todas las muestras digitales
Muestras análogas	0x00 0x00	Existen dos bytes para cada pin seleccionado como ADC
Suma de comprobación	0x99	Es la suma de toda la muestra API recibida

Extraído de (Digi) y modificado con datos obtenidos en experimento 1

Router 2

Tabla 6

Explicación de la muestra del router 2 con parámetro D0 "Low"

Nombre del campo	Muestra (HEX)	Descripción
Delimitador	0x7E	Todas las muestras poseen el mismo delimitador para conocer donde empieza cada paquete de datos recibido
Tamaño de la muestra	0x00 0x14	Tamaño de la muestra
Tipo de muestra	0x92	Identifica que la muestra fue automáticamente generada
64-bit, dirección de fuente	0x00 0x13 0xA2 0x00 0x40 0xE9 0x22 0xB7	Dirección del Xbee que envía la muestra
16-bit, dirección de fuente	0x23 0x57	Dirección del Xbee que envía la muestra
Opciones de recepción	0x01	Opciones de recepción
Número de muestras	0x01	Indica cuántas muestras fueron incluidas en el paquete
Digital channel mask	0x00 0x01	Indica los campos que fueron habilitados como DIO
Analog channel mask	0x08	Indica los campos que fueron habilitados como ADC
Muestras digitales	0x00 0x00	Contiene todas las muestras digitales
Muestras análogas	0x00 0x00	Existen dos bytes para cada pin seleccionado como ADC
Suma de comprobación	0x31	Es la suma de toda la muestra API recibida

Extraído de (Digi) y modificado con datos obtenidos en experimento 1

Tabla 7

Explicación de la muestra del router 2 con parámetro D0 "High"

Nombre del campo	Muestra (HEX)	Descripción
Delimitador	0x7E	Todas las muestras poseen el mismo delimitador para conocer donde empieza cada paquete de datos recibido
Tamaño de la muestra	0x00 0x14	Tamaño de la muestra
Tipo de muestra	0x92	Identifica que la muestra fue automáticamente generada
64-bit, dirección de fuente	0x00 0x13 0xA2 0x00 0x40 0xE9 0x22 0xB7	Dirección del Xbee que envía la muestra
16-bit, dirección de fuente	0x23 0x57	Dirección del Xbee que envía la muestra
Opciones de recepción	0x01	Opciones de recepción
Número de muestras	0x01	Indica cuántas muestras fueron incluidas en el paquete
Digital channel mask	0x00 0x01	Indica los campos que fueron habilitados como DIO
Analog channel mask	0x08	Indica los campos que fueron habilitados como ADC
Muestras digitales	0x00 0x01	Contiene todas las muestras digitales
Muestras análogas	0x00 0x00	Existen dos bytes para cada pin seleccionado como ADC
Suma de comprobación	0x30	Es la suma de toda la muestra API recibida

Extraído de (Digi) y modificado con datos obtenidos en experimento 1

Experimento 2

En este experimento, se ubicó el módulo Xbee Router 2 (figura 17) a una distancia de 42.5 m del Router 1 (figura 16), y esta a su vez a una distancia de 54 m del coordinador como se puede apreciar en la figura 13.

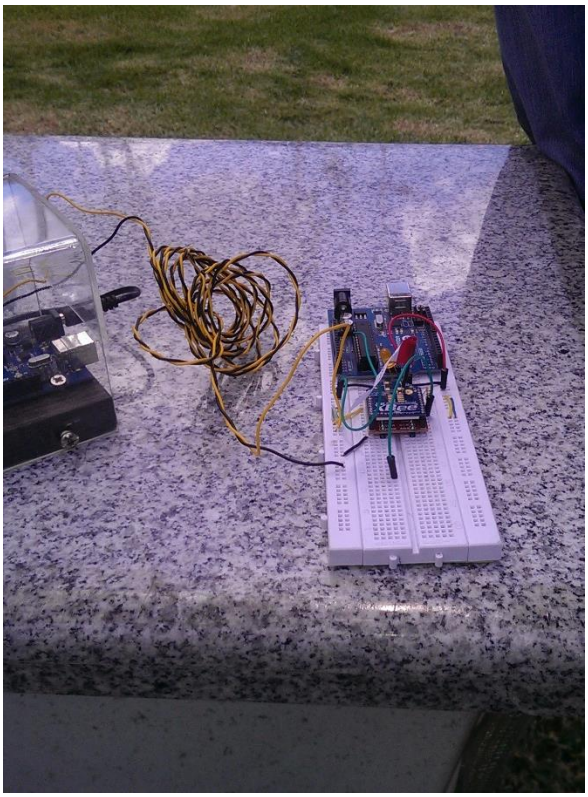

Figura 16. Router 1

A una distancia de 96.5 m desde el Xbee coordinador al Xbee router 2 (Figura 17), se intentó enviar y recibir datos sin que el Xbee router 1 esté encendido. Como resultado, no se obtuvo transferencia de datos entre estos dos módulos.

Posterior a esto, se encendió el Xbee router 1, que se encuentra aproximadamente en la mitad de los otros dos módulos, obteniendo como resultado, una transmisión de datos estable

entre los módulo Xbee router 2 y 1 con el Xbee coordinador, comprobando así, la topología tipo Cluster Tree.

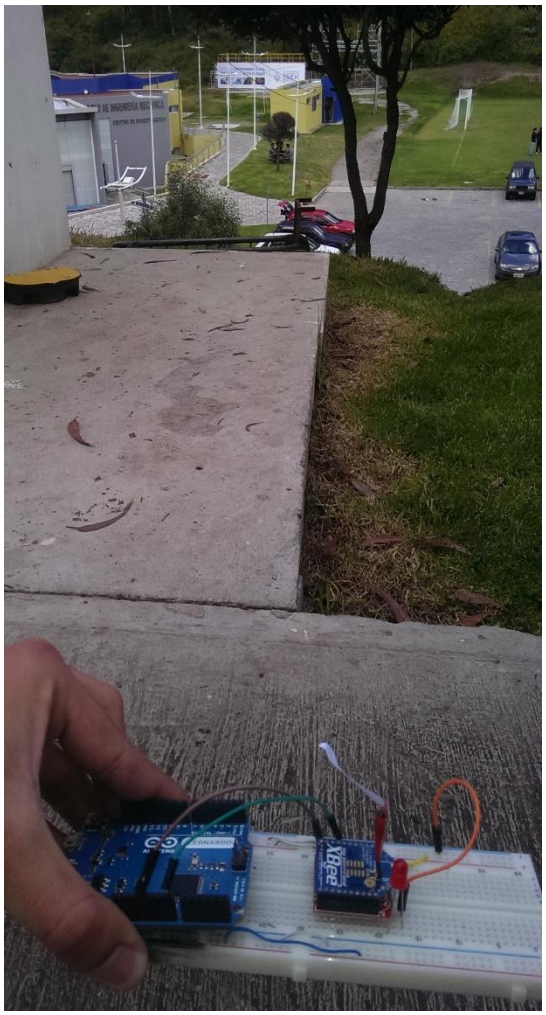

Figura 17. Router 2

Experimento 3

Los datos obtenidos son:

- Intensidad de módulos Xbee conectados a la red, con un promedio de 0,0191 A, como se puede apreciar en la figura 18

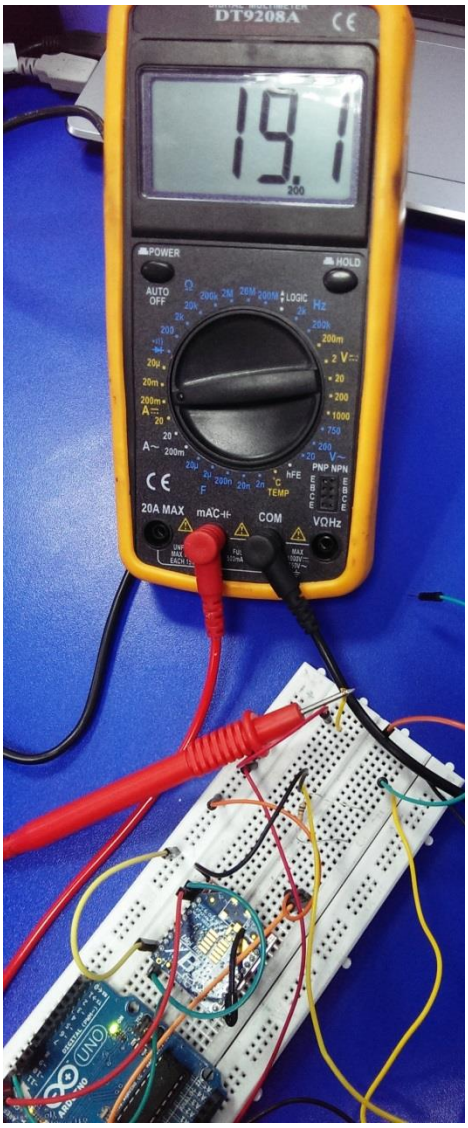

Figura 18. Intensidad de corriente de módulo Xbee S2

- Intensidad de módulos Xbee enviando y recibiendo datos, con un promedio de 0,0212 A como se puede apreciar en la figura 19.

Figura 19. Intensidad de corriente módulo Xbee S2 enviando datos

Ecuación de Potencia eléctrica

De acuerdo a los datos obtenidos, se procede a calcular la potencia cuando se están recibiendo datos en el módulo Xbee Coordinador, como se puede apreciar en la ecuación (6) y cuando se envían datos a los módulos Router, como se puede ver en la ecuación (7)

$$P = V * I \quad (5)$$

Dónde:

P = Potencia (W)

V = Voltaje (V)

I = Intensidad de corriente (A)

Cálculo de potencia consumida cuando el Xbee se encuentra recibiendo datos:

$$P = 3.3\ V * 0.0191\ A = 0{,}063\ W \quad (6)$$

Cálculo de potencia consumida cuando el Xbee se encuentra enviando datos:

$$P = 3.3\ V * 0.0212\ A = 0.07\ W \quad (7)$$

Discusión

Durante el desarrollo de este proyecto, se realizaron 3 experimentos en los cuales se comprobó el correcto desempeño de los módulos Xbee S2.

Dentro del experimento 1, se realizó la programación de los módulos Xbee utilizando el software X-Ctu. En este software, se asignó la jerarquía de cada Xbee y el valor que identificará a la red, evitando de esta manera posibles interferencias con otros equipos que también utilicen radiofrecuencia. Se recomienda abrir el programa X-ctu como administrador si se desea actualizar el firmware de los Xbee y también se debe establecer un "Sample time" diferente de cero en la configuración de los módulos Xbee Router, ya que de lo contrario no existiría recepción de datos en un Sample time = 0. Posterior a esto, se realizó la conexión de los módulos Xbee a las luminarias y a los sensores de corriente para analizar los paquetes de datos entrantes, donde se pudo comprobar que la muestra digital y la suma de comprobación son los únicos parámetros que se modificarán al encender o apagar las luminarias, manteniéndose el mismo número de bytes en cada paquete. Este fue un dato que se debía conocer antes de desarrollar la programación en Arduino, la cual va a reconocer los valores del sensor de corriente receptados de los Xbee Router.

En el experimento 2, se realizaron pruebas de red, verificando la jerarquía de cada módulo Xbee junto con el correcto funcionamiento de la interfaz desarrollada en Processing que enviará la trama de datos para encender o apagar las luminarias y que al mismo tiempo recibirá datos del sensor de corriente. Para este experimento, se verificó que no exista línea de vista directa entre el Xbee cordinador y el Xbee Router 2, siendo necesario que la información pase por el Xbee Router 1, comprobando de esta manera que la red es multipunto. La distancia máxima comprobada en este experimento fue de 54 m entre el Xbee coordinador y el Xbee Router 1.

En el experimento 3, se puso énfasis al consumo de energía de los módulos Xbee en varios estados. Se pudo concluir que el consumo energético de los módulos Xbee es 0.063W al recibir datos, y aumenta 0,007 W al enviar órdenes de encender o apagar las luminarias.

Se recomienda, sustituir el divisor de voltaje de la salida del sensor de corriente con diodos Zener para obtener un mayor rango y una mejor precisión en los datos recolectados o usar circuitos diseñados con amplificadores operacionales.

En trabajos futuros, se puede realizar modulación PWM utilizando micro controladores en cada nodo de la red Zigbee, para reducir la cantidad de luz de las luminarias cuando no sea posible logrando de esta forma un ahorro de energía adicional.

Referencias

Allegro MicroSystems. (s.f.). *Allegro*. Recuperado el 2016 de Junio de 29, de Allegro: http://www.allegromicro.com/~/media/Files/Datasheets/ACS712-Datasheet.ashx

Castillo, F., Sánchez, G., Ropero, J., Rivera, O., Benjumea, J., Barbancho, J., & Romero, M. (210). Servicios en Red. En F. Castillo, G. Sánchez, J. Ropero, O. Rivera, J. Benjumea, J. Barbancho, & M. Romero, *Servicios en Red* (pág. 230). Madrid: Paraninfo, SA.

Circutor. (s.f.). *Circutor.* Recuperado el 25 de Mayo de 2016, de Circutor: http://circutor.com/docs/CT_CIRLAMP_SP.pdf

Comisión de planificación y nomenclatura. (2003 de Marzo de 31). *Normas de Arquitectura y Urbanismo.* Recuperado el 10 de Mayo de 2016, de Normas de Arquitectura y Urbanismo: http://www7.quito.gob.ec/mdmq_ordenanzas/Ordenanzas/ORDENANZAS%20A%C3%91OS%20ANTERIORES/ORD-3457%20-%20NORMAS%20DE%20ARQUITECTURA%20Y%20URBANISMO.pdf

Digi. (2014). *Digi*. Recuperado el 22 de 3 de 2016, de Digi: http://ftp1.digi.com/support/documentation/html/90001399/90001399_A/Files/XBee-concepts.html

Digi. (s.f.). *Knowledge.digi*. Recuperado el 28 de 4 de 2016, de Knowledge.digi: http://knowledge.digi.com/articles/Knowledge_Base_Article/Digital-and-analog-sampling-using-XBee-radios

Digi. (s.f.). *Knowledge.digi.* Recuperado el 20 de Mayo de 2016, de Knowledge.digi: http://knowledge.digi.com/articles/Knowledge_Base_Article/Using-Remote-AT-Commands-to-Toggle-an-IO-on-a-Remote-XBee-ZB-Radio

El comercio. (7 de Enero de 2010). Expertos aconsejan bajar la velocidad. *El comercio*.

Fairchild. (11 de 2014). *Fairchildsemi*. Obtenido de Fairchildsemi: https://www.fairchildsemi.com/datasheets/TI/TIP31C.pdf

Farahani, S. (2008). ZigBee Wireless Networks and Transceivers. En S. Farahani, *ZigBee Wireless Networks and Transceivers* (pág. 1). Massachusetts: Elsevier Ltd.

La Hora. (10 de Abril de 2012). La falta de alumbrado afecta la seguridad en la Gonzáles Suárez. *La Hora*.

MaxStream. (01 de Junio de 2007). *Farnell.* Recuperado el 11 de Mayo de 2016, de Farnell: http://www.farnell.com/datasheets/27606.pdf

Montero, I. B. (2014). Instalación y mantenimiento de redes para transmisión de datos. En I. B. Montero, *Instalación y mantenimiento de redes para transmisión de datos* (pág. 66). Madrid: Paraninfo, SA.

Sáez, M. A. (2011). Crisis y reforma de la Administración púiblica. En M. A. Sáez, *Crisis y reforma de la Administración púiblica* (pág. 297). La Coruña: Gesbiblo, S.L.

Shorter, M. (s.f.). *Sparkfun Electronics.* Recuperado el 26 de 4 de 2016, de Sparkfun Electronics: https://cdn.sparkfun.com/assets/resources/2/9/20XbeePresentation.pdf

Sparkfun. (2006). *Sparkfun*. Obtenido de Sparkfun:

https://www.sparkfun.com/datasheets/BreakoutBoards/0712.pdf

Anexo A

Código de programación Arduino

```
#include <SoftwareSerial.h>
SoftwareSerial gtSerial(10, 11); // Arduino RX, Arduino TX
String inputString = "";
boolean stringComplete = false;

void setup()
{
Serial.begin(19200);
gtSerial.begin(9600);
}
void loop()
{
//*****************************Recepción de datos*********************************
if (gtSerial.available() >= 23) {
if (gtSerial.read() == 0x7E) {
int cero = gtSerial.read();

for (int j = 0; j<8; j++) { // Skip ahead to the analog data
byte discardByte = gtSerial.read();
}

int add = gtSerial.read();

//*****************************Xbee Router 1*********************************
if (add == 0x21){
for (int i = 11; i<21; i++) { // Skip ahead to the analog data
byte discardByte = gtSerial.read();
}
int analogMSB = gtSerial.read(); // Read the first analog byte data
int analogLSB = gtSerial.read(); // Read the second byte
int analogReading = analogLSB + (analogMSB * 256);
int envio = analogReading/4;
Serial.write(0x7E);
Serial.write(envio);
}
//*****************************Xbee Router 2*********************************
if (add == 0x22){
for (int i = 11; i<21; i++) { // Skip ahead to the analog data
byte discardByte1 = gtSerial.read();
```

```
}
int analogMSB = gtSerial.read(); // Read the first analog byte data
int analogLSB = gtSerial.read(); // Read the second byte
int analogReading = analogLSB + (analogMSB * 256);
int envio2 = analogReading/4;
//Serial.print("Dato 2 = ");
//Serial.println(analogReading);
Serial.write(0x7F);
Serial.write(envio2);
}
}
}



//*****************************Menú para encendido o apagado de las
luminarias**********************************
if (stringComplete)
{
inputString.trim();
if(inputString=="1.")
{
encender();
}
if(inputString=="2.")
{
apagar();
}
if(inputString=="3.")
{
encender1();
}
if(inputString=="4.")
{
apagar1();
}
inputString = "";
stringComplete = false;
}
}

void serialEvent() {
while (Serial.available()) {
//Recibir nueva variable
char inChar = (char)Serial.read();
//Agregarla a imputString
inputString += inChar;
```

```
if (inChar == '.') {
stringComplete = true;
}
}
}
//*****************************Encender Xbee Router 1*******************************
void encender()
{
gtSerial.write(0x7E);
gtSerial.write((byte)0x0);
gtSerial.write(0x10);
gtSerial.write(0x17);
gtSerial.write((byte)0x0);
gtSerial.write((byte)0x0);
gtSerial.write(0X13);
gtSerial.write(0XA2);
gtSerial.write((byte)0x0);
gtSerial.write(0X40);
gtSerial.write(0XE9);
gtSerial.write(0x21);
gtSerial.write(0XD7);
gtSerial.write(0XFF);
gtSerial.write(0XFE);
gtSerial.write(0X02);
gtSerial.write('D');
gtSerial.write('0');
gtSerial.write(0X05);
long chexsum1 = (0x17 + 0x00 + 0x13 + 0xA2 + 0x00 + 0x40 + 0xE9 + 0x21 + 0xD7 +0xFF +
0xFE + 0x02 + 'D' + '0' + 0x05);
gtSerial.write( 0xFF - (chexsum1 & 0xFF));
//Serial.println("Se encendio el LED");

}
//*****************************Apagar Xbee Router 1*******************************
void apagar()
{
gtSerial.write(0x7E);
gtSerial.write((byte)0x0);
gtSerial.write(0x10);
gtSerial.write(0x17);
gtSerial.write((byte)0x0);
gtSerial.write((byte)0x0);
gtSerial.write(0X13);
gtSerial.write(0XA2);
gtSerial.write((byte)0x0);
gtSerial.write(0X40);
gtSerial.write(0XE9);
```

```
gtSerial.write(0x21);
gtSerial.write(0XD7);
gtSerial.write(0XFF);
gtSerial.write(0XFE);
gtSerial.write(0X02);
gtSerial.write('D');
gtSerial.write('0');
gtSerial.write(0X04);
long chexsum1 = (0x17 + 0x00 + 0x13 + 0xA2 + 0x00 + 0x40 + 0xE9 + 0x21 + 0xD7 +0xFF +
0xFE + 0x02 + 'D' + '0' + 0x04);
gtSerial.write( 0xFF - (chexsum1 & 0xFF));
//Serial.println("Se apago el LED");
}
//*****************************Encender Xbee Router 2*******************************
void encender1()
{
gtSerial.write(0x7E);
gtSerial.write((byte)0x0);
gtSerial.write(0x10);
gtSerial.write(0x17);
gtSerial.write((byte)0x0);
gtSerial.write((byte)0x0);
gtSerial.write(0X13);
gtSerial.write(0XA2);
gtSerial.write((byte)0x0);
gtSerial.write(0X40);
gtSerial.write(0XE9);
gtSerial.write(0x22);
gtSerial.write(0XB7);
gtSerial.write(0XFF);
gtSerial.write(0XFE);
gtSerial.write(0X02);
gtSerial.write('D');
gtSerial.write('0');
gtSerial.write(0X05);
long chexsum1 = (0x17 + 0x00 + 0x13 + 0xA2 + 0x00 + 0x40 + 0xE9 + 0x22 + 0xB7 +0xFF +
0xFE + 0x02 + 'D' + '0' + 0x05);
gtSerial.write( 0xFF - (chexsum1 & 0xFF));
//Serial.println("Se encendio el LED 2");
}
//*****************************Apagar Xbee Router 1*******************************
void apagar1()
{
gtSerial.write(0x7E);
gtSerial.write((byte)0x0);
gtSerial.write(0x10);
gtSerial.write(0x17);
```

```
gtSerial.write((byte)0x0);
gtSerial.write((byte)0x0);
gtSerial.write(0X13);
gtSerial.write(0XA2);
gtSerial.write((byte)0x0);
gtSerial.write(0X40);
gtSerial.write(0XE9);
gtSerial.write(0x22);
gtSerial.write(0XB7);
gtSerial.write(0XFF);
gtSerial.write(0XFE);
gtSerial.write(0X02);
gtSerial.write('D');
gtSerial.write('0');
gtSerial.write(0X04);
long chexsum1 = (0x17 + 0x00 + 0x13 + 0xA2 + 0x00 + 0x40 + 0xE9 + 0x22 + 0xB7 +0xFF +
0xFE + 0x02 + 'D' + '0' + 0x04);
gtSerial.write( 0xFF - (chexsum1 & 0xFF));
//Serial.println("Se apago el LED 2");
}
```

Anexo B

Código de programación Processing

```
import processing.serial.*; //Se Importa la librería Serial

Serial port; //Nombre del puerto serie
//Encender router 1
int rquad=30; //Radio del quadrado
int xquad=100;  //Posición X de rect
int yquad=200;  //Posición Y de rect
boolean overRect = false; //Estado del mouse si está encima de rect o no
//Apagar router 1
int rquad1=30; //Radio del quadrado
int xquad1=100;  //Posición X de rect
int yquad1=300;  //Posición Y de rect
boolean overRect1 = false; //Estado del mouse si está encima de rect o no
//Encender router 2
int rquad2=30; //Radio del quadrado
int xquad2=500;  //Posición X de rect
int yquad2=200;  //Posición Y de rect
boolean overRect2 = false; //Estado del mouse si está encima de rect o no
//Apagar router 2
int rquad3=30; //Radio del quadrado
int xquad3=500;  //Posición X de rect
int yquad3=300;  //Posición Y de rect
boolean overRect3 = false; //Estado del mouse si está encima de rect o no
//Colores del boton router 1
int red=100;
int green=100;
int blue=100;
//Colores del boton router 1
int red1=100;
int green1=100;
int blue1=100;
//Colores del boton router 2
int red2=100;
int green2=100;
int blue2=100;
//Colores del boton router 2
int red3=100;
int green3=100;
int blue3=100;
boolean status=false; //Estado del color de rect
boolean status1=false; //Estado del color de rect
boolean status2=false; //Estado del color de rect
```

```
boolean status3=false; //Estado del color de rect
int xlogo=425;//Posición X de la imagen
int ylogo=60;//Posición Y de la imagen
int valor;
int dato1;
int dato2;



void setup()
{
  println(Serial.list()); //Visualiza los puertos serie disponibles en la consola de abajo
  port = new Serial(this, Serial.list()[0], 19200); //Abre el puerto serie
    size(850, 400); //Se crea una ventana de los pixeles descritos
}

void draw()
{
  background(255,255,255);//Fondo de color blanco

  if(mouseX > xquad-rquad && mouseX < xquad+rquad &&  //Si el mouse se encuentra dentro
de rect
     mouseY > yquad-rquad && mouseY < yquad+rquad)
     {
       overRect=true;  //Variable que demuestra que el mouse esta dentro de rect
     }
   else
   {
     overRect=false;  //Si el mouse no está dentro de rect, la variable pasa a ser falsa
   }

    if(mouseX > xquad1-rquad1 && mouseX < xquad1+rquad1 &&  //Si el mouse se encuentra
dentro de rect
     mouseY > yquad1-rquad1 && mouseY < yquad1+rquad1)
     {
       overRect1=true;  //Variable que demuestra que el mouse esta dentro de rect
     }
   else
   {
     overRect1=false;  //Si el mouse no está dentro de rect, la variable pasa a ser falsa
   }

    if(mouseX > xquad2-rquad2 && mouseX < xquad2+rquad2 &&  //Si el mouse se encuentra
dentro de rect
     mouseY > yquad2-rquad2 && mouseY < yquad2+rquad2)
     {
       overRect2=true;  //Variable que demuestra que el mouse esta dentro de rect
```

```
    }
  else
  {
   overRect2=false;  //Si el mouse no está dentro de rect, la variable pasa a ser falsa
  }

  if(mouseX > xquad3-rquad3 && mouseX < xquad3+rquad3 &&  //Si el mouse se encuentra
dentro de rect
    mouseY > yquad3-rquad3 && mouseY < yquad3+rquad3)
    {
     overRect3=true;  //Variable que demuestra que el mouse esta dentro de rect
    }
  else
  {
   overRect3=false;  //Si el mouse no está dentro de rect, la variable pasa a ser falsa
  }

 //Se dibuja un rectangulo de color gris
  fill(red,green,blue);
  rectMode(RADIUS); //Esta función hace que Width y Height de rect sea el radio (distancia
desde el centro hasta un costado).
  rect(xquad,yquad,rquad,rquad);

  fill(red1,green1,blue1);
  rectMode(RADIUS); //Esta función hace que Width y Height de rect sea el radio (distancia
desde el centro hasta un costado).
  rect(xquad1,yquad1,rquad1,rquad1);

  fill(red2,green2,blue2);
  rectMode(RADIUS); //Esta función hace que Width y Height de rect sea el radio (distancia
desde el centro hasta un costado).
  rect(xquad2,yquad2,rquad2,rquad2);

  fill(red3,green3,blue3);
  rectMode(RADIUS); //Esta función hace que Width y Height de rect sea el radio (distancia
desde el centro hasta un costado).
  rect(xquad3,yquad3,rquad3,rquad3);


  //Se ubica el logo en la pantalla
 imageMode(CENTER);//Esta función hace que las coordenadas de la imagne sean el centro de
esta y no la esquina izquierda arriba
 PImage imagen=loadImage("logo.png");
 image(imagen,xlogo,ylogo,200,100);


  if(port.available() > 0) // si hay algún dato disponible en el puerto
```

```
  {
    valor=port.read();//Lee el dato y lo almacena en la variable "valor"
    if (valor==0x7E){
      dato1 = port.read();
    }
    if (valor==0x7F){
      dato2 = port.read();

          }
  }
    int uno = dato1*4;

  //Se visualiza el dato del router 1
  fill(50);
  text("Dato de Xbee 0013A20040E921D7",190,250);
  text(uno, 270, 270);
  //Se visualiza el dato del router 2
   int dos = dato2*4;
  fill(50);
  text("Dato de Xbee 0013A20040E922B7",590,250);
  text(dos, 670, 270);
  }

void mousePressed()  //Cuando el mouse está apretado
{
 //***************Encender Luminaria Router 1***************
 if (overRect==true) //Si el mouse está dentro de rect
 {
   status=!status; //El estado del color cambia
   port.write("1."); //Envia una "A" para que el Arduino encienda el led
   if(status==true)
   {
     red=0;
     green=255;
     blue=0;
     red1=100;
     green1=100;
     blue1=100;
   }

   if(status==false)
   {
     red=0;
     green=255;
     blue=0;
     red1=100;
     green1=100;
```

```
    blue1=100;

  }

 }
 //***************Apagar Luminaria Router 1***************
 if (overRect1==true) //Si el mouse está dentro de rect
 {
  status1=!status1; //El estado del color cambia
  port.write("2."); //Envia una "A" para que el Arduino encienda el led
  if(status1==true)
  {
   //Color del botón rojo
   red1=255;
   green1=0;
   blue1=0;
   red=100;
   green=100;
   blue=100;

  }

  if(status1==false)
  {
   //Color del botón negro
   red1=255;
   green1=0;
   blue1=0;
   red=100;
   green=100;
   blue=100;

  }

 }
 //***************Encender Luminaria Router 2***************
 if (overRect2==true) //Si el mouse está dentro de rect
 {
  status2=!status2; //El estado del color cambia
  port.write("3."); //Envia una "A" para que el Arduino encienda el led
  if(status2==true)
  {
   //Color del botón rojo
   red2=0;
   green2=255;
   blue2=0;
   red3=100;
```

```
      green3=100;
      blue3=100;

    }

    if(status2==false)
    {
      //Color del botón negro
      red2=0;
      green2=255;
      blue2=0;
      red3=100;
      green3=100;
      blue3=100;

    }

  }
  //***************Apagar Luminaria Router 1***************
   if (overRect3==true) //Si el mouse está dentro de rect
  {
    status3=!status3; //El estado del color cambia
    port.write("4."); //Envia una "A" para que el Arduino encienda el led
    if(status3==true)
    {
      //Color del botón rojo
      red3=255;
      green3=0;
      blue3=0;
      red2=100;
      green2=100;
      blue2=100;

    }

    if(status3==false)
    {
      //Color del botón negro
      red3=255;
      green3=0;
      blue3=0;
      red2=100;
      green2=100;
      blue2=100;
    }
  }
}
```

Anexo C

Interfaz creada en Processing

Anexo D

Características eléctricas obtenida de (MaxStream, 2007)

Symbol	Parameter	Condition	Min	Typical	Max	Units
VIL	Input Low Voltage	All Digital Inputs	-	-	0.2 * VCC	V
V_{IH}	Input High Voltage	All Digital Inputs	0.8 * VCC	-	0.18* VCC	V
V_{OL}	Output Low Voltage	I_{OL} = 2 mA, VCC >= 2.7 V	-	-	0.18*VCC	V
V_{OH}	Output High Voltage	I_{OH} = -2 mA, VCC >= 2.7 V	0.82*VCC	-	-	V
I_{IN}	Input Leakage Current	V_{IN} = VCC or GND, all inputs, per pin	-	-	0.5uA	uA
TX	Transmit Current	VCC = 3.3 V	-	45	-	mA
RX	Receive Current	VCC = 3.3 V	-	50	-	mA
PWR-DWN	Power-down Current	SM parameter = 1	-	< 10	-	uA