



UNIVERSIDAD TECNOLÓGICA ISRAEL

TRABAJO DE TITULACIÓN EN OPCIÓN AL GRADO DE:

**“INGENIERO EN ELECTRÓNICA DIGITAL Y
TELECOMUNICACIONES”**

TEMA:

**DISEÑO Y DESARROLLO DE ETIQUETAS ELECTRÓNICAS INALÁMBRICAS
PARA PRODUCTOS EN SUPERMERCADOS, MEDIANTE TECNOLOGÍA
ZIGBEE.**

AUTOR:

SANTIAGO DAVID VILLAMARÍN AGUIRRE.

TUTOR:

MSC. RENÉ ERNESTO CORTIJO LEYVA

QUITO, ECUADOR

2018

UNIVERSIDAD TECNOLÓGICA ISRAEL

APROBACIÓN DEL TUTOR

En mi calidad de tutor del componente práctico certifico:

Que el trabajo de titulación “**DISEÑO Y DESARROLLO DE ETIQUETAS ELECTRÓNICAS INALAMBRICAS PARA PRODUCTOS EN SUPERMERCADOS, MEDIANTE TECNOLOGÍA ZIGBEE.**”, presentado por la Sr. Santiago David Villamarín Aguirre, estudiante de la carrera de Electrónica Digital y Telecomunicaciones, reúne los requisitos y méritos suficientes para ser sometido a la evaluación del Tribunal de Grado, que se designe, para su correspondiente estudio y calificación.

Quito D.M. Agosto del 2018

TUTOR

.....
Ing. Rene Ernesto Cortijo Leyva, Mg

AGRADECIMIENTOS.

Expreso mi total agradecimiento a todas las personas que me han ayudado y apoyado en la realización de este proyecto, en especial a mis padres que sin sus consejos y apoyo no hubiera sido posible.

También un especial agradecimiento a mis profesores de la Universidad Tecnológica Israel, que me supieron enseñar las bases para poder realizar este proyecto, en especial al Mg. Rene Cortijo que con su guía pude terminar satisfactoriamente este proyecto.

DEDICATORIA.

El presente proyecto de investigación está dedicado a mis padres que supieron apoyarme y me incentivaron a creer que es posible terminar una parte muy importante en mi vida estudiantil.

A esta prestigiosa Universidad que me permitió continuar con mis estudios.

A mis profesores que me enseñaron a parte de los conocimientos, valores que los llevare en toda mi vida profesional.

ÍNDICE DE CONTENIDO

1 INTRODUCCIÓN.....	1
Antecedentes de la situación del objeto estudiado.....	1
1.1 Planteamiento del problema.....	1
1.2 Formulación del problema.....	1
1.3 Justificación.....	2
1.4 Objetivo General.....	2
1.4.2 Objetivos específicos.....	2
2 CAPÍTULO I. FUNDAMENTACIÓN TEÓRICA.....	3
2.1 TECNOLOGÍA ZIGBEE.....	3
2.1.2 Características:	4
2.1.3 TOPOLOGÍAS DE RED EN EL ESTÁNDAR ZIGBEE.....	5
2.1.4 Comunicaciones Básicas.....	6
2.1.5 Beacon-habilitado vs. Red NonBeacon.....	6
2.1.6 Transferencia de datos.....	7
2.1.7 Transferencia de datos a un coordinador desde un dispositivo.....	7
2.1.8 Transferencia de datos de un coordinador a un dispositivo	8
2.1.9 Transferencia de datos entre dispositivos iguales.....	8
2.1.10 Funciones de la capa de red Zigbee o 802.15.4	8
2.1.11 Estructura de trama.....	9
2.1.12 Seguridad.....	10
2.1.13 Zigbee frente a otras tecnologías.....	11
2.2 Microcontrolador ATmega328P.....	11
3 CAPÍTULO II. PROPUESTA.....	13
3.1 Descripción del Sistema.....	13
3.2 Diseño del hardware.....	13
3.2.2 Coordinador principal.....	13
3.2.3 Etiqueta Electrónica.....	14
3.3 Diseño del software.....	15
3.3.4 Programación del Microcontrolador ATM328P.....	15

3.3.5 Programación de la Aplicación en Visual Basic.	16
3.4 Análisis de costos y tiempo requerido del proyecto.	16
3.5 Ventajas de las etiquetas electrónicas.	18
4 CAPÍTULO III. IMPLEMENTACIÓN	19
4.1 Hardware	19
Introducción	19
4.1.2 Diseño de la controladora principal.	19
4.1.3 Diseño de las etiquetas electrónicas.	20
4.2 Software.	21
4.2.4 Programación de la controladora central.....	21
4.2.5 Programación de las etiquetas electrónicas.	23
4.2.6 Programación de la aplicación para el usuario.....	24
4.3 Pruebas de funcionamiento.	26
4.4 Análisis de resultados.	26
5 CONCLUSIONES Y RECOMENDACIONES.	28
5.1 CONCLUSIONES.	28
Al finalizar el proyecto se tiene las siguientes conclusiones	28
5.2 RECOMENDACIONES.....	29
6 REFERENCIAS BIBLIOGRÁFICAS.	30
7 ANEXOS:	32

ÍNDICE DE GRÁFICOS.

Figura 2.1: Capas de protocolo de red inalámbrica Zigbee.....	3
Figura 2.2: Topología de red de IEEE 802.15.4.....	5
Figura 2.3: Transferencia de datos a un coordinador a) Beacon habilitado b) Beacon no habilitado.	7
Figura 2.4: Transferencia de datos de un coordinador (a)con Beacon (b) sin Beacon.	8
Figura 2.5: Estructura de paquetes Zigbee.	9
Figura 2.6: Estructura de MAC Beacon.	9
Figura 3.1: Diagrama electrónico de coordinador principal.	14
Figura 3.2: Diagrama electrónico de etiqueta electrónica.....	15
Figura 3.3: Flujograma inicial de configuración de la aplicación desarrollada en Visual Basic 6.0	16
Figura 4.1: Diseño del diagrama PCB de la controladora central.....	19
Figura 4.2: Implementación controladora principal.....	20
Figura 4.3: Diseño de diagrama PCB de las etiquetas electrónicas.	20
Figura 4.4: Placas de las etiquetas electrónicas.....	21
Figura 4.5: Librerías y variables de la controladora principal.	22
Figura 4.6: Programa para envío de letras controladora principal.	22
Figura 4.7: Librería de las etiquetas electrónicas.....	23
Figura 4.8: Variables LCD etiquetas electrónicas.....	23
Figura 4.9: Recepción y almacenamiento de letras en la memoria EEPROM.....	24
Figura 4.10: Programa Visual Basic utilizado.	24
Figura 4.11: Programación en Visual Basic 6.0.....	25
Figura 4.12: Presentación de la aplicación para el usuario.	25
Figura 7.1: Pantalla etiqueta electrónica	79
Figura 7.2: Pantalla etiqueta electrónica.	79

ÍNDICE DE TABLAS.

Tabla 2.1: Parámetros del microcontrolador ATmega328P.....	12
Tabla 3.1: Costo total de materiales del proyecto.	16
Tabla 3.2: Costo total de materiales del proyecto.	17
Tabla 3.3: Costo de materiales de la tarjeta electrónica.	17
Tabla 3.4: Tiempo requerido en la realización del proyecto.	18
Tabla 4.1: Pruebas de alcance de etiquetas electrónicas.	26
Tabla 4.2: Pruebas de pérdida de datos sin voltaje.....	26
Tabla 4.3: Pruebas de batería en etiquetas electrónicas.	26

ÍNDICE DE ANEXOS.

Anexo 1: Precios de elementos electrónicos usados.	32
Anexo 2: Xbee en el mercado.	33
Anexo 3: Data sheet AT MEGA 328P	36
Anexo 4: Programación etiqueta electrónica en Arduino IDE.....	38
Anexo 5: Cronograma de actividades.	70
Anexo 6: Diagrama electrónico de la controladora pincipal.....	72
Anexo 7: Diagrama etiqueta Electronica	73
Anexo 8: MANUAL DE USUARIO	74
Anexo 9: MANUAL TÉCNICO	81

RESUMEN.

El presente proyecto se basa en la utilización de la tecnología Zigbee, para la realización de etiquetas electrónicas para supermercados, el objetivo es mejorar la forma que se las realiza actualmente y presentar de una forma más llamativa la descripción y costo del producto.

El funcionamiento del proyecto es mediante una controladora central que está conectada a un PC. Esta controladora en su hardware está conformada por un módulo Zigbee y un microcontrolador ATM 328P. La controladora será la encargada de comunicarse con las etiquetas electrónicas vía ondas electromagnéticas. Para cambiar la descripción del producto, precio y oferta se desarrolló un programa en Visual 6.0 que será instalado en el PC.

Las etiquetas electrónicas en su hardware están conformadas por un módulo Zigbee, un microcontrolador ATM 328P y un display GLCD. La información como descripción del producto, precio y oferta, llega de la controladora central a la etiqueta electrónica y esta cambia automáticamente sus valores.

PALABRAS CLAVES: microcontrolador ATM 328P, Zigbee, GLCD, Visual 6.0, controladora central.

ABSTRACT.

The present project is based on the use of Zigbee technology for the realization of electronic labels for supermarkets, the objective is to improve the way they are currently carried out and to present in a more striking way the description and cost of the product.

The operation of the project is through a central controller that is connected to a PC. This controller in its hardware, is formed by a Zigbee module and an ATM 328P microcontroller. The controller will be in charge of communicating with electronic tags via electromagnetic waves. To change the description of the product, price and offer, a Visual 6.0 program was developed that will be installed on the PC.

The electronic labels in its hardware are composed of a Zigbee module, an ATM 328P microcontroller and a GLCD display. The information as a description of the product, price and offer, comes from the central controller to the electronic label and this automatically changes its values.

KEYWORDS: ATM microcontroller 328P, Zigbee, GLCD, Visual 6.0, central controller.

1 INTRODUCCIÓN.

Antecedentes de la situación del objeto estudiado.

En los supermercados del Ecuador para dar información del producto, tanto en precio y descripción, se utilizan etiquetas echas de polipropileno o cartón usando impresoras industriales. Estas etiquetas se cambian regularmente ya que los precios varían semanalmente debido a cambios y promociones, los trabajadores tienen que digitar en la computadora la descripción y el precio para luego imprimirla. Esto se hace diariamente, dedicando tiempo del trabajador y recursos de la empresa.

Para mejorar el tiempo que se usa para cambiar etiquetas y optimizar recursos en Supermercados, se va a diseñar un sistema que consta un dispositivo inalámbrico y etiquetas electrónicas con tecnología Zigbee. Para ello vamos a cumplir con los siguientes objetivos.

Se realizará un estudio de la tecnología Zigbee.

Determinar el lenguaje de programación adecuado para el diseño.

1.1 Planteamiento del problema.

En el Ecuador los supermercados no han implementado etiquetas electrónicas para dar información al cliente sobre el producto a comprar. Con las etiquetas electrónicas ayudaríamos a ahorrar tiempo y recursos de los Supermercados. Ya que estas se pueden cambiar vía inalámbrica, conforme a las necesidades del trabajador.

1.2 Formulación del problema.

Determinación de los elementos y la mejor tecnología que sirve para realizar nuestras etiquetas electrónicas para Supermercados.

1.3 Justificación.

Se realizarán etiquetas electrónicas con tecnología Zigbee, porque en el mercado ecuatoriano su implementación es muy reducida y tiene grandes ventajas en comparación a las etiquetas que se usan hoy en día. Se las va a construir para facilitar el manejo de la información del producto tanto del cliente como el trabajador del Supermercado. La investigación se la realizará observando que se encuentra en el mercado y con qué elementos contamos para realizar nuestra etiqueta.

1.4 Objetivo General.

Implementar etiquetas electrónicas inalámbricas para productos en supermercados, mediante tecnología Zigbee.

1.4.2 Objetivos específicos.

- Definir las técnicas, parámetros y componentes electrónicos para la implementación de etiquetas inalámbricas electrónicas, usando tecnología Zigbee.
- Diseñar el circuito de control, visualización y comunicación de la etiqueta electrónica inalámbrica.
- Diseñar una aplicación que controle remotamente la información mostrada en las etiquetas electrónicas.
- Implementar la etiqueta electrónica de acuerdo con el diseño realizado.
- Realizar pruebas de la implementación y el control de los resultados obtenidos.

2 CAPÍTULO I. FUNDAMENTACIÓN TEÓRICA

2.1 TECNOLOGÍA ZIGBEE

Existen diferentes tecnologías que se encargan de la transmisión de datos a pequeño y largo alcance como WIFI, bluetooth utilizadas en redes LAN, WPAN. Pero existía la necesidad de una tecnología para uso exclusivo de control de sensores de mediano alcance que cumpla con características como bajo consumo de voltaje, baja latencia, pequeña memoria, bajo ancho de banda. Por ello se creó el protocolo Zigbee que es un conjunto de normas, para manejar Zigbee.

En la figura 2.1 se observa cómo está determinado el protocolo Zigbee, la capa física y de acceso está determinada por el estándar IEE 802.15.4 y se encarga de manejar la entrada y salida de datos. Zigbee está diseñado para dar cabida a circuitería análoga que permita implementaciones de bajo costo. La capa de control de acceso Mac y la capa de subsoporte de aplicación APS se encargan de la seguridad del estándar Zigbee. (Farahani, 2008: 4)

La capa de aplicación está abierta y corre a cargo de cada fabricante.

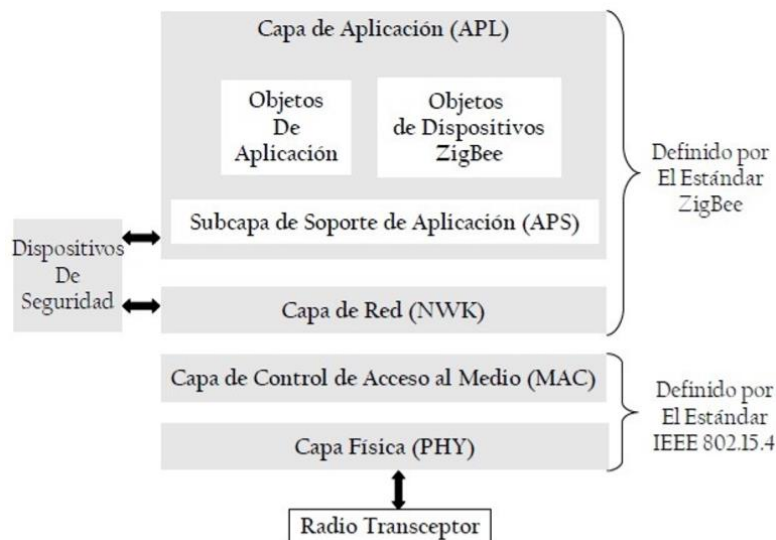


Figura 2.1: Capas de protocolo de red inalámbrica Zigbee.

Fuente: (Farahani, 2008)

Zigbee está trabajando en estas bandas:

868–868.6 MHz (868 MHz band)

902–928 MHz (915 MHz band)

2400–2483.5 MHz (2.4 GHz band)

La banda de 868 MHz es usada en Europa por aplicaciones, incluyendo red inalámbrica de corto alcance.

Las bandas de 915MHz y 2.4 GHz son parte de la industria científica y médica.

La banda de 915 MHz es usada mayormente en EE. UU.

La banda de 2.4 GHz es usada por todo el mundo. (Farahani, 2008:8)

2.1.2 Características:

- La interfaz USB permite la comunicación y la programación de los dispositivos.
- Transferencia de datos confiable a través de la encriptación con MAC.
- Bajo consumo de energía ya que tiene un modo de sueño (transmisión/recepción).
- Alta densidad de módulos y nodos por red para facilidad de interconexiones.
- Bajo costo de instalación y bajo costo de mantenimiento.
- Protocolo simple que facilita la implementación global.
- No requiere gran cantidad de memoria FLASH ni ROM.
- Esta echo para aplicaciones de bajo ciclo de actividad ($\leq 0.1\%$).
- Baja potencia duración de batería por años. (Gutierrez, 2015)

2.1.3 TOPOLOGÍAS DE RED EN EL ESTÁNDAR ZIGBEE

De acuerdo con la especificación IEEE 802.15.4, el LR-WPAN puede operar en una de las dos topologías de red: de estrella o de punto a punto. IEEE 802.15.4 está diseñado para redes con baja velocidad de datos (Chonggang et al., 2014:42).

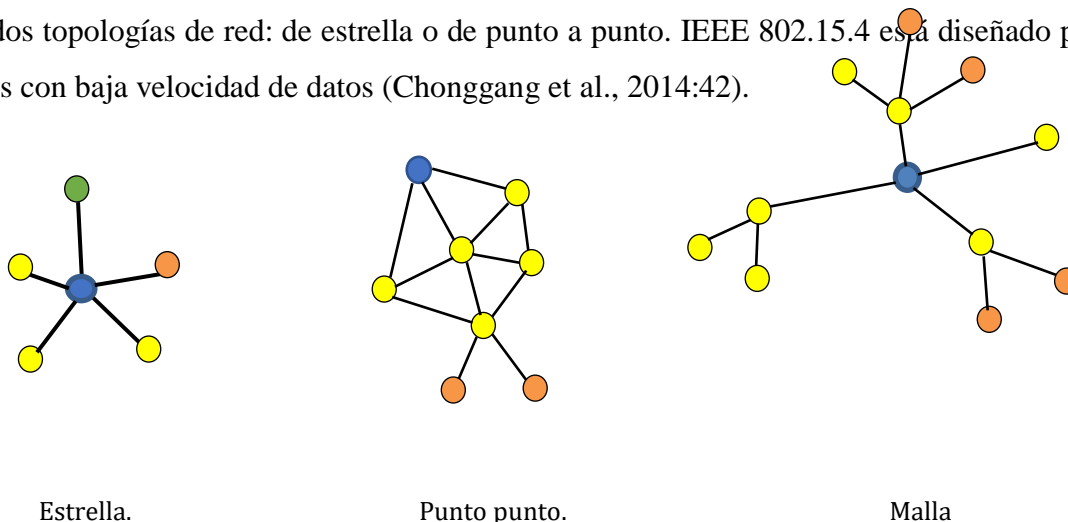


Figura 2.2: Topología de red de IEEE 802.15.4
Fuente: (Elaborado por el autor)

- PAN Coordinador.
- Dispositivo función completa.
- Dispositivo función reducida.

Para menor costo del sistema hay dos tipos de dispositivos: dispositivos de función completa (FFD) y dispositivos de función reducida.

. Topología Estrella.

En la topología estrella figura 2.2, todos los dispositivos de la red pueden comunicarse solo con el coordinador de PAN. Un escenario típico en una formación de red estelar es que una FFD (Full Function Device), programada para ser un coordinador de PAN, se activa y comienza a establecer su red. Lo primero que hace este coordinador de PAN es seleccionar un identificador de PAN único que no sea utilizado por ninguna otra red en su radio de influencia: la región alrededor del dispositivo en la que su radio puede comunicarse con éxito con otras radios. En otras palabras, garantiza que el identificador PAN no sea utilizado por ninguna otra red cercana (Chonggang et al., 2014:43).

El coordinador de PAN controla la red y realiza los siguientes deberes mínimos:

- Asigne una dirección única (16 bits o 64 bits) a cada dispositivo de la red.
- Inicia, finaliza y enruta los mensajes en toda la red.
- Solo existe un único coordinador PAN por Red.

.Topología punto a punto.

La red punto a punto figura 2.2, puede tener varias restricciones para comunicarse entre sí, si no tiene restricciones se denomina malla.

Otra forma que admite la tecnología Zigbee es la topología árbol, en la cual existe un coordinador de PAN que establece la condición inicial. Tiene dispositivos finales que actúan como hojas de árbol y no participan en el enrutamiento. Los enrutadores forman ramas y retransmiten los mensajes. Los enrutadores hacen crecer la red Zigbee (Chonggang et al., 2014:43).

2.1.4 Comunicaciones Básicas.

IEEE 802.15.4 utiliza métodos para que múltiples dispositivos pueda transmitir en un solo canal. El mecanismo de acceso al canal utilizado es el Acceso Múltiple con sentido de operador con prevención de colisiones (CSMA-CA).

El método detección de energía ED, se utiliza para detectar que el canal está ocupando la intensidad de señal, por lo que sólo mide el nivel de energía y determina si es o no una señal 802.15.4.

Otro método usado es el sentido de portadora CS en el cual se detecta si el canal está ocupado y si esta usado por una señal 802.15.4. Si esta usado por una señal 802.15.4 transmite así esté la señal por debajo del nivel de energía. Si no está claro que está ocupado el canal, apaga el dispositivo y lo intenta más tarde, considerando el número de veces que haya sido programado.

2.1.5 Beacon-habilitado vs. Red NonBeacon.

Existen 2 métodos para el acceso a canales basados en contención y libre de contención. En el de contención se transmiten con el mecanismo CSMA-CA que descubren un canal

disponible y transmiten. En un segundo método libre de contención el coordinador PAN asigna un tiempo garantizado (GTS,) el dispositivo comenzará a transmitir en este tiempo. En este método Beacon es un mensaje con formato específico que asegura que los relojes estén sincronizados en los nodos de la red.

Para sincronizar relojes se utiliza un método de baliza, que es un paquete de bits usado para saber que todos los dispositivos estén conectados al coordinador. La desventaja de este método es que los dispositivos deben recibir periódicamente la señal de baliza despertarse sincronizar la señal de reloj y volver a dormir. Sin tener que transmitir información, por tanto, el tiempo de batería es menor en este método (Farahani, 2008:13).

2.1.6 Transferencia de datos.

Hay tres tipos de transferencia de datos.

- Transferencia de datos a un coordinador desde un dispositivo
- Transferencia de datos de un coordinador a un dispositivo
- Transferencia de datos entre dos dispositivos iguales punto a punto.

2.1.7 Transferencia de datos a un coordinador desde un dispositivo

En la Figura 2.3(a) se observa la transmisión de datos con Beacon en el cual se sincronizan los relojes y se transmiten por el método CSMA-CA (Farahani, 2008:14). En la figura 2.3(b) Beacon está deshabilitado, existe la transferencia de datos en cuanto la red esté habilitada. Acuse de recibo es opcional en los dos métodos.

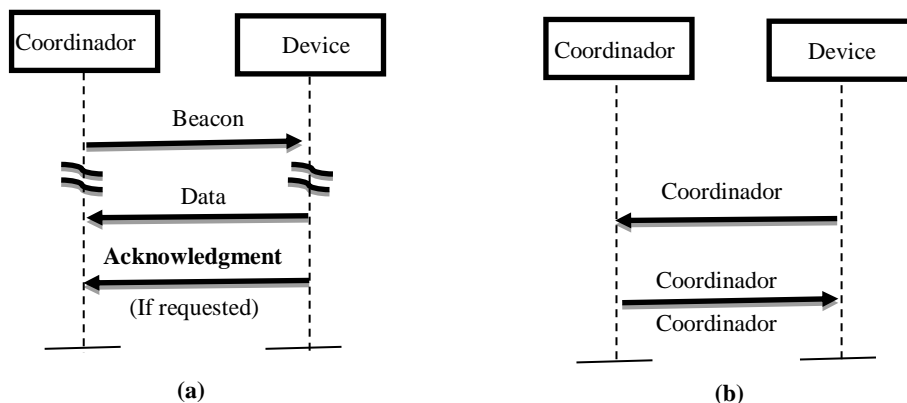


Figura 2.3: Transferencia de datos a un coordinador a) Beacon habilitado b) Beacon no habilitado.

Fuente: (Farahani, 2008)

2.1.8 Transferencia de datos de un coordinador a un dispositivo

En la figura 2.4 se puede observar los pasos que se tiene que seguir para enviar información desde el coordinador hacia el dispositivo con y sin Beacon. Si el coordinador necesita enviar datos envía un mensaje de Beacon al dispositivo, este a su vez envía un mensaje de confirmación que está listo y activo para recibir los datos (Farahani, 2008:16). El coordinador envía un mensaje de recibido y envía los datos. El acuse de recibo es opcional.

En la red no habilitada para control remoto, el coordinador espera que el dispositivo envíe un mensaje para ver si tiene datos pendientes, el coordinador da un mensaje de recibido y luego transmite los datos.

2.1.9 Transferencia de datos entre dispositivos iguales.

En una topología punto a punto, cada dispositivo se puede comunicar directamente con cualquier otro dispositivo.

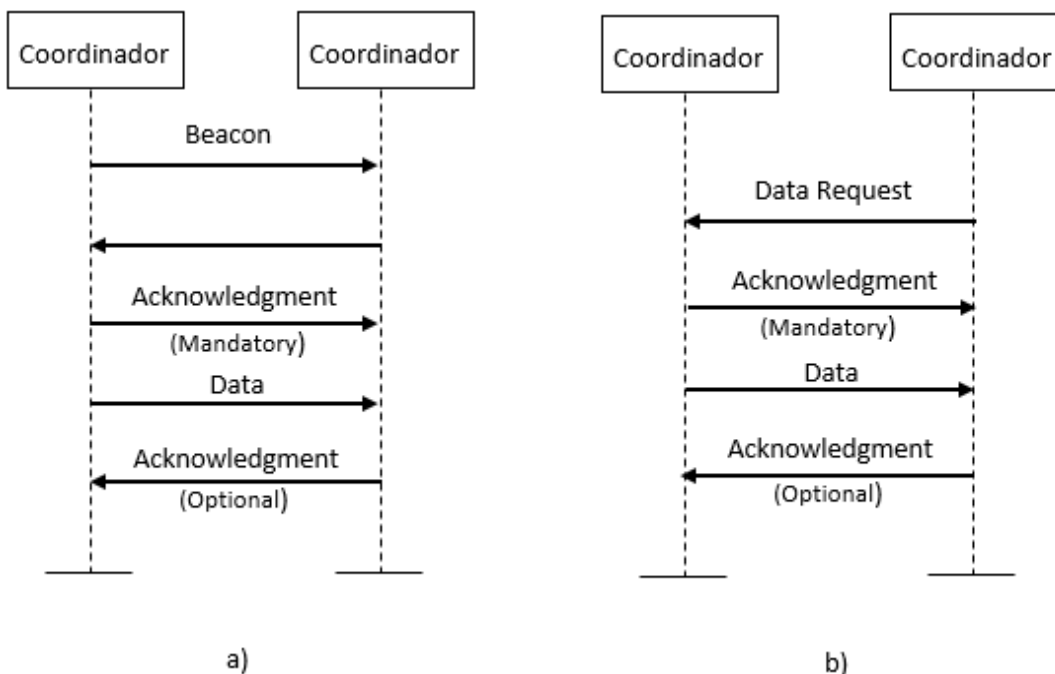


Figura 2.4: Transferencia de datos de un coordinador (a) con Beacon (b) sin Beacon.
Fuente: (Farahani, 2008)

2.1.10 Funciones de la capa de red Zigbee o 802.15.4

En la Fig.2.5 se observa la capa PHY o 802.15.4, es la capa más cercana al hardware y

directamente controla y comunica el transceptor de radio.

La capa PHY se encarga de activar la transmisión y recepción de datos (Farahani, 2008:17). También selecciona la frecuencia y se encarga de asegurar que el canal no esté siendo ocupado para poder transmitir.

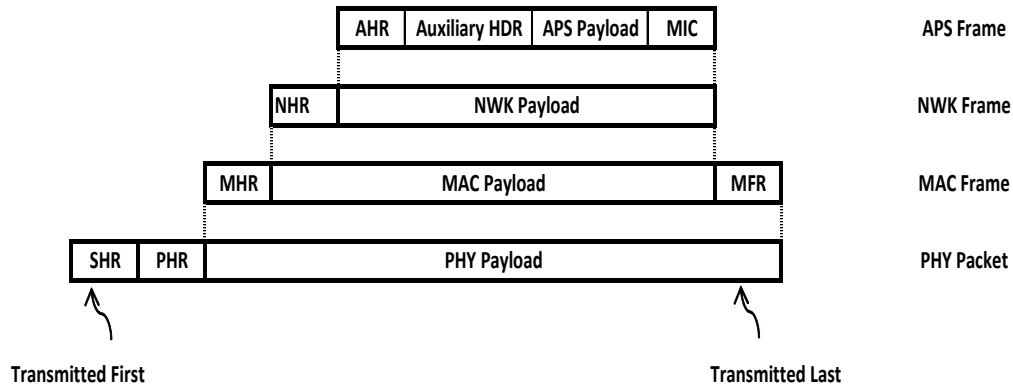


Figura 2.5: Estructura de paquetes Zigbee.
Fuente: (Farahani, 2008)

2.1.11 Estructura de trama.

La capa Mac se encarga de generar balizas y sincronizar los equipos con las balizas. Se definen cuatro estructuras de trama MAC:

- Marco de baliza. Se usan para generar balizas y sincronizar relojes.
- Marco de datos. Se usan para transmitir datos
- Marco de acuse de recibo. Información de aviso de recibo.
- Marco de comando MAC. Transmiten el comando MAC.

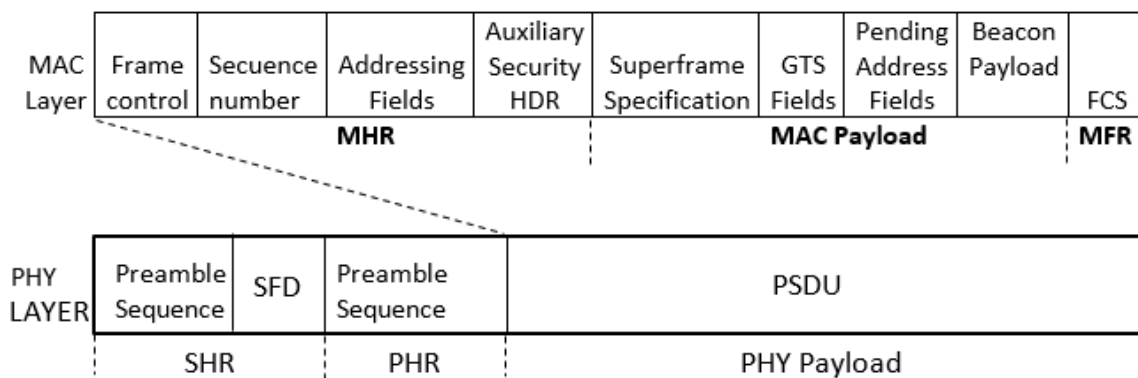


Figura 2.6: Estructura de MAC Beacon.
Fuente: (Farahani, 2008)

En la figura 2.6 se tiene la estructura de capas MAC, la trama MAC completa se usa como una carga en un paquete PHY. PHY se define como unidad de datos de servicio. El campo preámbulo es utilizado para la sincronización.

SFD es un delimitador que nos indica el final de SHR y el inicio PHR. La longitud PSDU (carga útil PHY) se determina por el número total de octetos.

El marco MAC consta de tres secciones: el encabezado MAC (MHR), la carga MAC y el pie de página MAC (MFR). El campo de control de cuadro en el MHR contiene información que define el tipo de cuadro, los campos de direccionamiento y otros indicadores de control. El número de secuencia especifica el número de secuencia de baliza (BSN). El campo de direccionamiento proporciona las direcciones de origen y destino. El encabezado de seguridad auxiliar es opcional y contiene la información requerida para el procesamiento de seguridad.

2.1.12 Seguridad.

En la red inalámbrica, los mensajes transmitidos pueden recibirse en cualquier dispositivo cercano, incluido un intruso. Hay dos preocupaciones principales de seguridad en una red inalámbrica. El primero es la confidencialidad de los datos. El dispositivo intruso puede obtener información sensible simplemente escuchando los mensajes transmitidos. Encriptar los mensajes antes de la transmisión resolverá el problema de confidencialidad. El estándar IEEE 802.15.4 admite el uso de Advanced Encryption Standard (AES) [14] para encriptar sus mensajes salientes. La segunda preocupación es que el dispositivo intruso puede modificar y volver a enviar uno de los mensajes anteriores, incluso si los mensajes están encriptados. Con un código de integridad de mensaje (MIC) con cada trama saliente le permitirá al destinatario saber si el mensaje ha sido cambiado en tránsito. Este proceso se conoce como autenticación de datos. Una de las principales limitaciones para implementar funciones de seguridad en una red inalámbrica Zigbee es la limitación de recursos. Los nodos son alimentados principalmente por baterías y tienen un poder computacional y un tamaño de memoria limitados. Zigbee está dirigido a aplicaciones de bajo costo y el hardware en los nodos podría no ser resistente a alteraciones. Si un intruso adquiere un nodo de una red operativa que no tiene resistencia a la manipulación, la clave real podría obtenerse simplemente de la memoria del dispositivo. Un nodo a prueba de

manipulaciones puede borrar la información confidencial, incluidas las claves de seguridad, si se detecta una manipulación (Farahani, 2008:22).

2.1.13 Zigbee frente a otras tecnologías.

Bluetooth vs Zigbee.

La diferencia está en el enfoque hacia su aplicación.

Bluetooth está enfocada a la voz incorporando un sistema de salto de frecuencia con un protocolo maestro esclavo. Zigbee está enfocado a sensores, controles y aplicaciones con mensajes cortos, mediante sistemas de secuencia directa con protocolo estrella o punto (Kinney, 2003:16).

Diferencias:

Zigbee puede tener un máximo de 65535 nodos distribuidos en subredes de 255 nodos frente a los 8 máximos en una red bluetooth. Zigbee tiene menor consumo eléctrico, tiene 30mA transmitiendo y 3uA en reposo. Bluetooth 40 mA transmitiendo y 0.2 mA en reposo. Zigbee pasa casi todo el tiempo en reposo en cambio bluetooth esta siempre transmitiendo.

Zigbee tiene una velocidad máxima de 250Kbps en tanto Bluetooth tiene 1Mbps (Kinney, 2003:18).

Las propias tareas en las que se encargan cada uno por ejemplo Bluetooth está presente en aplicaciones móviles, informática casera. Zigbee es insuficiente para esta tarea siendo usado para domótica, productos dependientes de la batería, sensores médicos, juguetes electrónicos donde la transferencia de datos es menor

2.2 Microcontrolador ATmega328P.

Es un microcontrolador de la fábrica ATMEL basado en la arquitectura RISC (computadora con conjunto de instrucciones reducidas. Trabaja a una tensión de 1.8 a 5.5v, dispone de 28 pines de entrada/salida (Atmel, 2018:2). Tiene diferentes tipos de configuraciones dependiendo de la aplicación.

En la tabla 2.1 se observa las especificaciones técnicas del microcontrolador, que nos

ayudan a tener una visión general del mismo.

Tabla 2.1: Parámetros del microcontrolador ATmega328P.

Especificaciones Técnicas.	
Microcontroler	ATmega328P
Operating Voltage	5V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
Analog Input Pins	32 KB (ATmega328P) of which 0.5KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1KB(ATmega328P)
Clock Speed	16 Mhz

Fuente: (Sebastian, 2018)

3 CAPÍTULO II. PROPUESTA.

3.1 Descripción del Sistema.

En la actualidad se observa en los supermercados que colocan los precios en las perchas por medio de etiquetas de cartón, esto lo realizan diariamente. A veces las ponen unas encima de otras por falta de tiempo. Para solventar estos problemas hemos optado por el diseño e implementación de las tarjetas electrónicas. Para ello vamos a dividir nuestro sistema en dos etapas.

La primera etapa es un coordinador central, que va a ser el que nos permita enviar y recibir información de las etiquetas electrónicas. Esta etapa utiliza el microcontrolador AMT 328P, en el cual se realiza la comunicación de las etiquetas electrónicas con la aplicación de la PC.

La segunda etapa son las etiquetas electrónicas que reciben información del coordinador central y la muestran en el display GLCD.

3.2 Diseño del hardware.

3.2.2 Coordinador principal.

El coordinado principal está configurado en red estrella y es el denominado coordinador PAN. A él se conectan todas las etiquetas electrónicas. El coordinador principal consta de un convertidor USB-TTL, un módulo Zigbee S1 y el microcontrolador ATMega328P.

El convertidor USB-TTL permite la comunicación serial a puerto USB, por medio de el podemos comunicarnos sin problema con nuestra aplicación en una computadora y programar al ATMega328P. Para la transmisión y recepción de datos del Zigbee con el microcontrolador AT328, utilizamos los pines 2 y 3 del microcontrolador.

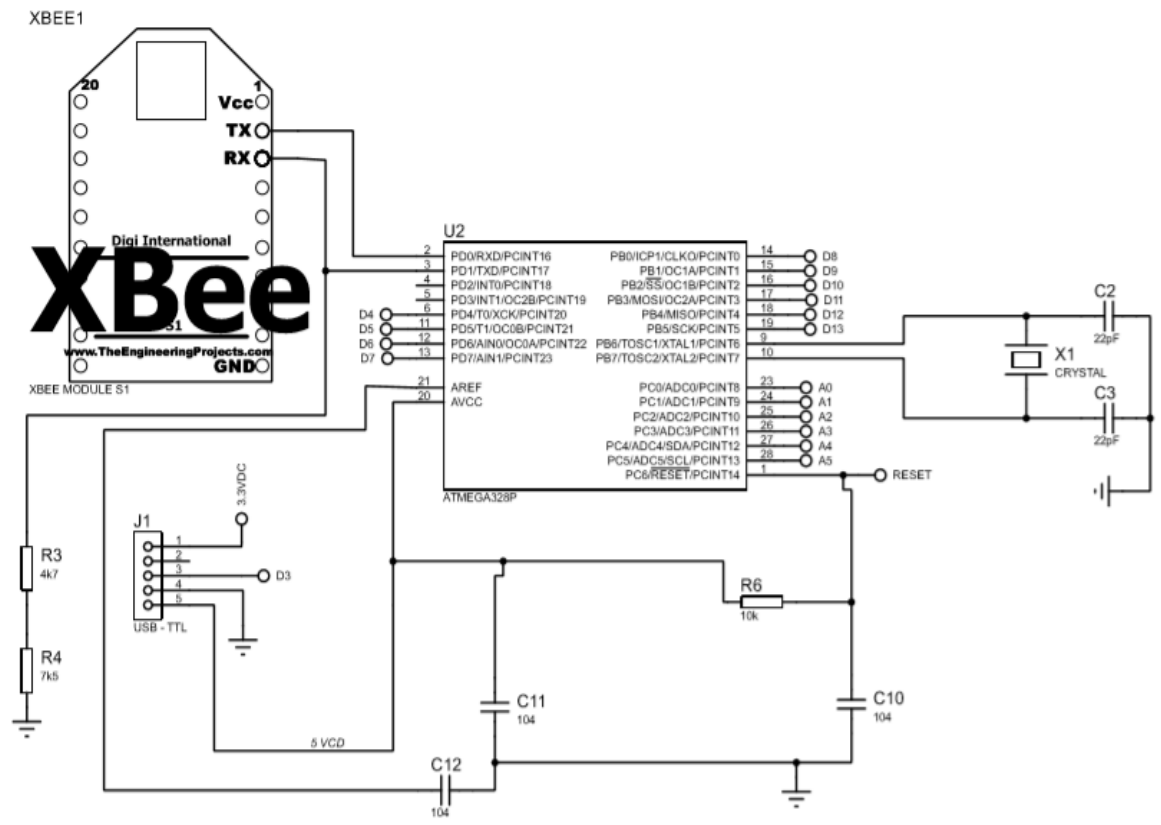


Figura 3.1: Diagrama electrónico de coordinador principal.
Fuente: (Elaborado por el autor).

3.2.3 Etiqueta Electrónica.

En la figura 3.2 se encuentra el diagrama de la etiqueta electrónica, está compuesta principalmente de un módulo Zigbee, el microcontrolador ATM328P, y el display gráfico GLCD.

El módulo Zigbee se asocia a la controladora principal y es el encargado de la transmisión y recepción de datos.

El microcontrolador se encarga de recibir la información por medio del módulo Zigbee, utiliza una señal de reloj con un cristal de cuarzo de 16000 MHz.

El display gráfico GLCD recibe la información del microcontrolador por medio de sus puertos digitales y los muestra al cliente.

3.3 Diseño del software.

3.3.4 Programación del Microcontrolador ATM328P.

La programación se la realizó en el programa Arduino IDE, ya que el microcontrolador AMT 328P, reconoce las instrucciones. Después de programar y con las pruebas en simulación, se carga la aplicación al microcontrolador por medio de una placa Arduino Uno. Se utilizó el lenguaje de programación Arduino que es un C++ adaptado que tiene una librería de alta calidad que utiliza un compilador de C++ para los microcontroladores AVR de Atmel. El programa IDE incluye librerías de forma automática y no es necesario tener que declararlas, pero en casos específicos como el nuestro si declaramos 2 librerías una para el convertidor USB-TTL y otra para nuestro GLCD.

```
//#include "Adafruit_GFX.h"// Hardware-specific library
```

```
//#include <Adafruit_TFTLCD.h>
```

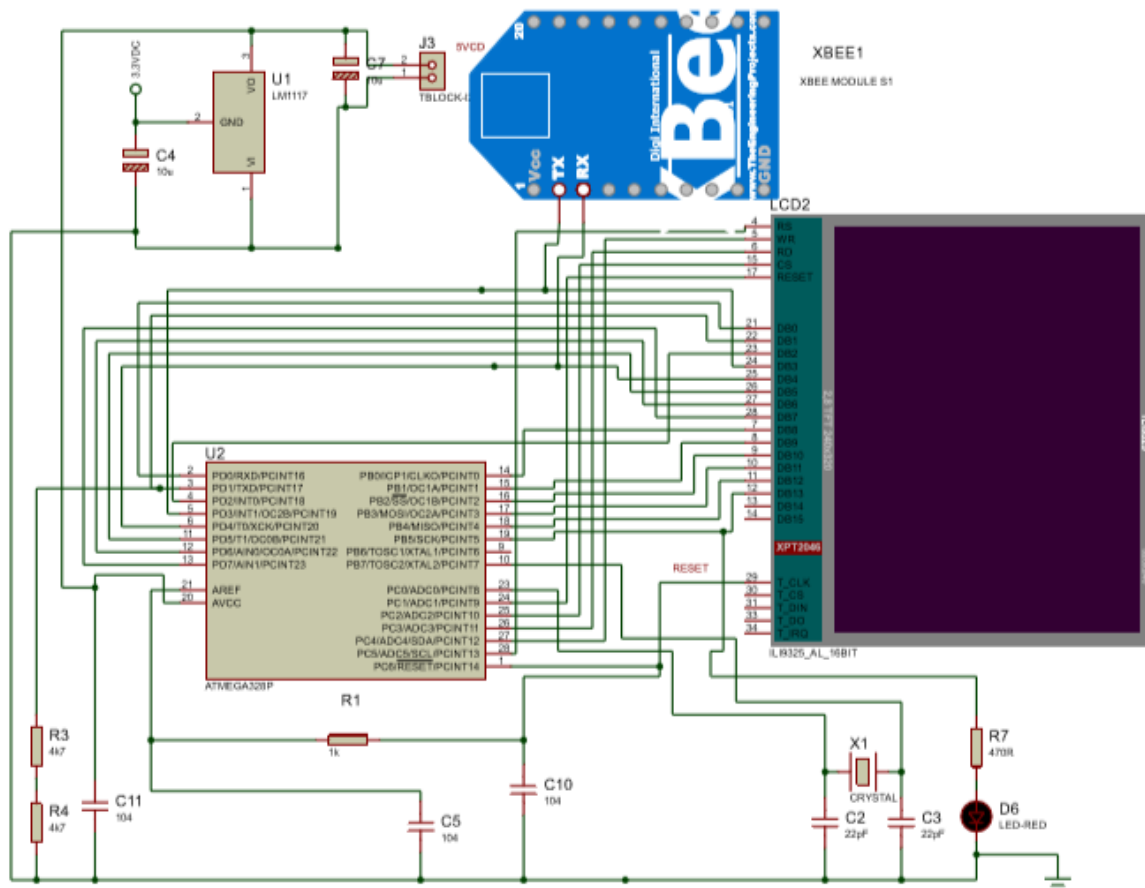


Figura 3.2: Diagrama electrónico de etiqueta electrónica.

Fuente: (Elaborado por el autor).

3.3.5 Programación de la Aplicación en Visual Basic.

La aplicación para modificar las etiquetas electrónicas esta realizado en Visual Basic 6.0 y se utiliza un convertidor USB a TTL. El convertidor nos permite comunicarnos a la controladora central que se encarga de enviar y recibir información de las etiquetas electrónicas.

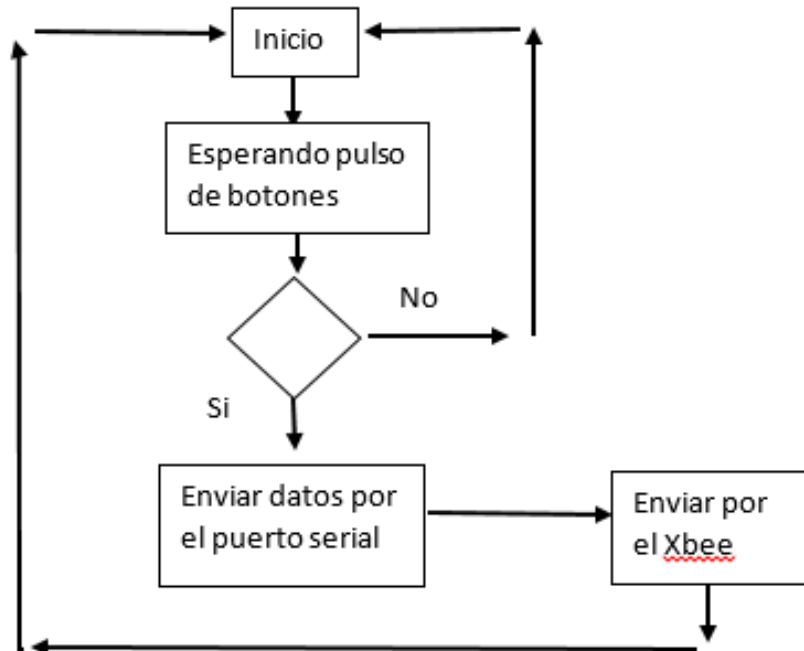


Figura 3.3: Flujograma inicial de configuración de la aplicación desarrollada en Visual Basic 6.0

Fuente: (Elaborado por el autor)

3.4 Análisis de costos y tiempo requerido del proyecto.

En la tabla 3.1 se encuentran los precios en el mercado y la cantidad de elementos electrónicos necesarios para la construcción del sistema de etiquetas electrónicas. El costo total del sistema etiquetas electrónicas con tecnología Zigbee es de 144,60 dólar.

Tabla 3.1: Costo total de materiales del proyecto.

Materiales	Precio (USD)	Cantidad	Precio total (USD)
Xbee USB Adapter	10	1	10
Módulo Lcd Tft para Arduino	10	2	20
Xbee S1	48	2	96

Fuente: (Elaborado por el autor)

Tabla 3.1: Costo total de materiales del proyecto.

Materiales	Precio (USD)	Cantidad	Precio total (USD)
Baquelita	4,5	1	1
Oscilador de Cuarzo 16Mhz	1	3	3
Resistencias 1/4W	0,2	15	3
Zocalos para baquelita	0,45	12	5,4
Regulador de voltage LM1117	1,5	2	3
Capacitor cerámico 104	0,2	8	1,6
Capacitor cerámico 10uf	0,2	2	0,4
Capacitor cerámico 22pf	0,2	6	1,2
Caja etiqueta electrónica	20	2	40
Caja controladora	20	1	20
Pila 4,2v 700mA	8	2	16
Módulo carga de batería	5,50	2	11
Costo Total del proyecto			231,6

Fuente: (Elaborado por el autor)

En la tabla 3.2 se tiene el costo por etiqueta electrónica, es necesario separar este costo ya que las etiquetas electrónicas se las tiene que hacer en mayor número.

Tabla 3.2: Costo de materiales de la tarjeta electrónica.

Materiales	Precio (USD)	Cantidad	Precio total (USD)
Xbee USB Adapter	10	1	10
Módulo Lcd TFT para Arduino	10	1	10
Xbee S1	48	1	48
Baquelita	4,5	1	1
Oscilador de Cuarzo 16Mhz	1	1	1
Resistencias 1/4W	0,2	5	1
Zocalos para baquelita	0,45	3	1,35
Regulador de voltage LM1117	1,5	1	1,5
Capacitor cerámico 104	0,2	2	0,4
Capacitor cerámico 22pf	0,2	6	1,2
Caja protectora de etiqueta	20	1	20
Costo por etiqueta electrónica			95,45

Fuente: (Elaborado por el autor)

Se realizó el estudio de costos a la etiqueta electrónica por separado ya que estas las realizará en un mayor número dependiendo de la cantidad de productos que se necesita

etiquetar.

En la tabla 3.3 se hizo un estudio del tiempo que se necesitó para la realización del proyecto y el costo por hora.

Tabla 3.3: Tiempo requerido en la realización del proyecto.

	Tiempo/h	Costo*hora/dólares	Total/dólares
Diseño del proyecto	64	4	256
Implementación del proyecto	152	4	608
Pruebas	36	4	144
Total	256		1008

Fuente: (Elaborado por el autor)

3.5 Ventajas de las etiquetas electrónicas.

- Son fáciles de instalar y modificar su información dependiendo de las necesidades del usuario.
- El costo de producción es menor a las que se encuentran en el Mercado.
- Su aplicación es de fácil manejo e instalación permitiendo facilidades en su implementación.
- Solo se requiere de un costo inicial, de ahí las etiquetas tienen larga duración. Por lo que el costo beneficio es una ventaja ante el método usado actualmente.
- Opera en banda libre de 2.4GHz, para redes inalámbricas.
- Gran facilidad de escalabilidad en usuarios y nodos.
- Bajo ciclo de trabajo, larga vida de la batería.
- Provee conexiones seguras con encriptación AES.

4 CAPÍTULO III. IMPLEMENTACIÓN

4.1 Hardware

Introducción

El proyecto tuvo 2 diseños de hardware: una controladora central y 2 etiquetas electrónicas. Se realizó el diagrama circuital de los dos diseños, con el que se realizaron pruebas por medio del programa proteus y la programación del microcontrolador ATM 328P.

4.1.2 Diseño de la controladora principal.

Se realizó el diagrama circuital de nuestra controladora principal, tomando en cuenta las protecciones tanto para nuestro microcontrolador ATM 328P como para el módulo Zigbee. Con la opción PCB del programa proteus figura 4.1 se realizó las pistas de PCB, para ser quemadas en la baquelita.

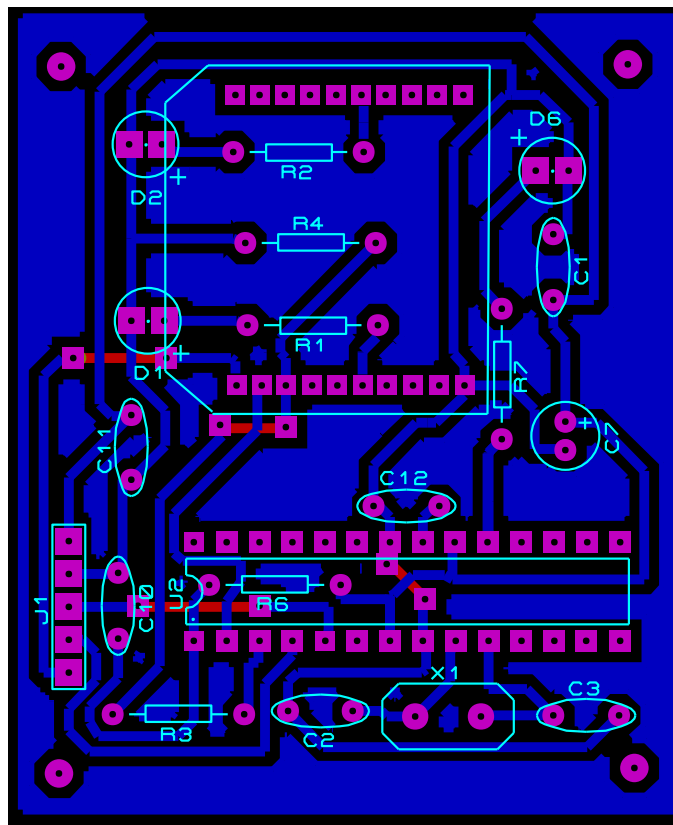


Figura 4.1: Diseño del diagrama PCB de la controladora central.
Fuente: (Elaborado por el autor).

Luego se procede a soldar los elementos electrónicos figura 4.2 que conforman nuestra controladora principal. Cuidando de no quemar los elementos y conectarlos correctamente.

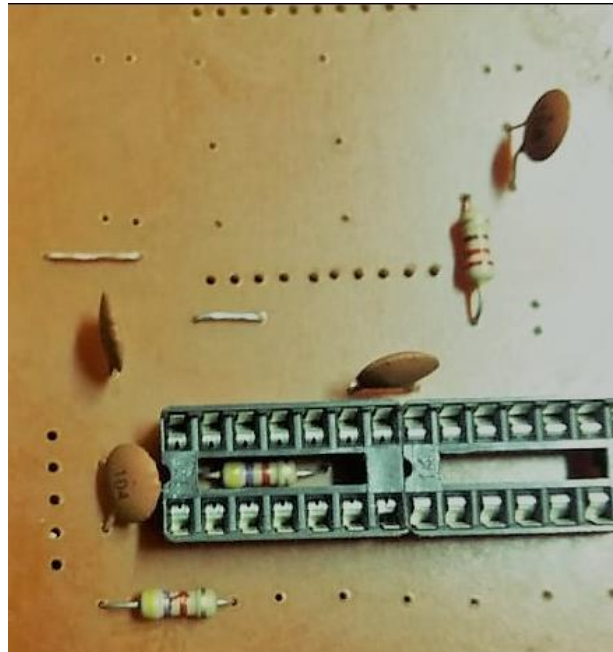


Figura 4.2: Implementación controladora principal.
Fuente: (Elaborado por el autor)

4.1.3 Diseño de las etiquetas electrónicas.

En la figura 4.3 se encuentra el diagrama circuitual de las etiquetas electrónicas, tomando en cuenta las protecciones tanto para nuestro microcontrolador ATM 328P como para el módulo Zigbee. Con la opción PCB del programa proteus se realizó las pistas de PCB, para ser quemadas en la baquelita.

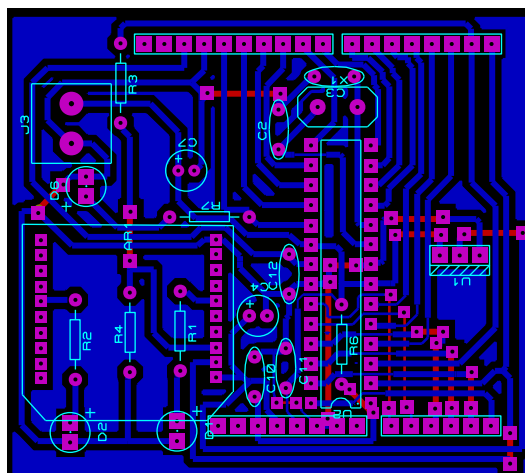


Figura 4.3: Diseño de diagrama PCB de las etiquetas electrónicas.
Fuente:(Elaborado por el autor)

Luego al diagrama se lo transfirió con papel transfer a una baquelita, se quemó la

baquelita con ácido clorhídrico, y se realizó la limpieza y pruebas de las pistas.

Si se obtiene una media de continuidad obtenida con el multímetro no está de acuerdo con el diseño, de debe separar esas pistas con ayuda de un estilete. Esto se realiza para evitar corto circuitos.

En la Fig. 4.4 se desarrolla el montaje de los elementos electrónicos para proceder a soldar cuidando de no dañar las pistas ni los elementos con el cautín.

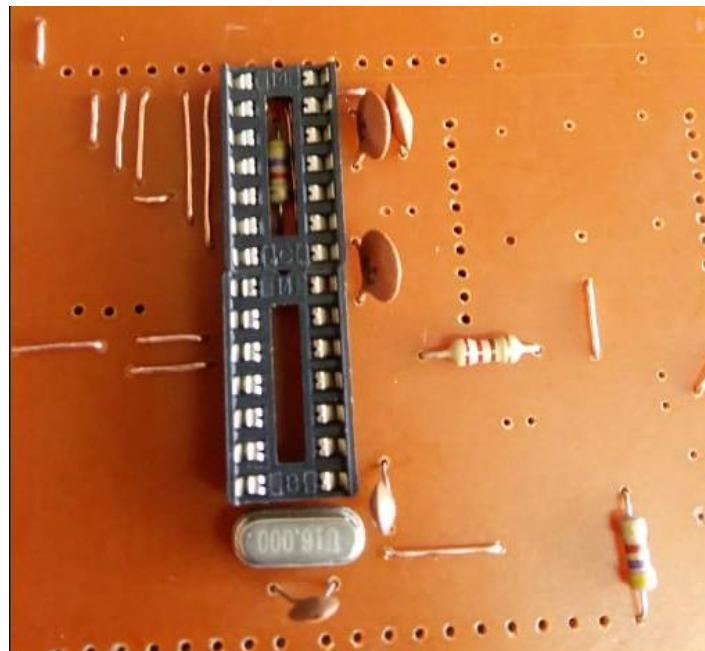


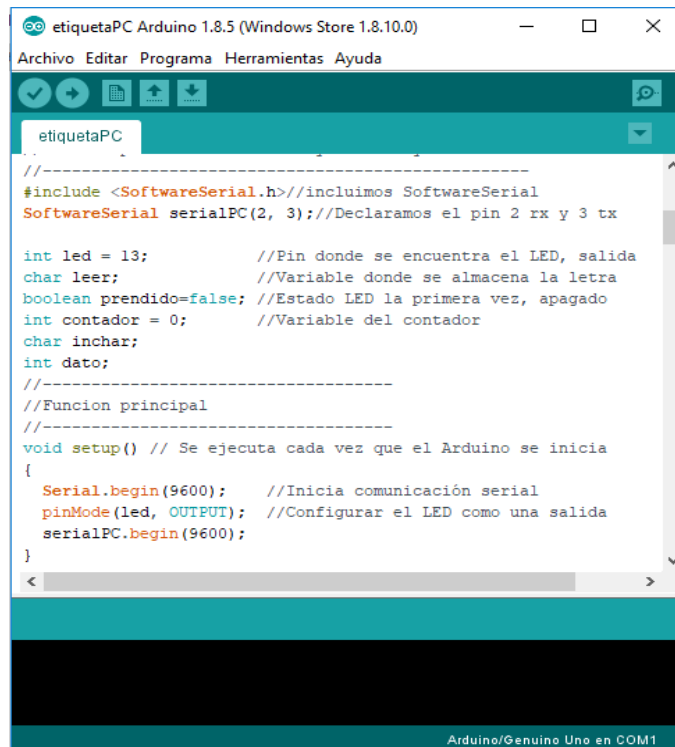
Figura 4.4: Placas de las etiquetas electrónicas.
Fuente: (Elaborado por el autor)

4.2 Software.

Para la parte del software del proyecto se van a utilizar dos aplicaciones. La programación de la aplicación que se lo realizó en Visual Basic y la programación de las etiquetas electrónicas y coordinadora principal en el programa Arduino IDE 1.8.5.

4.2.4 Programación de la controladora central.

Esta programación fue realizada utilizando el programa Arduino IDE. En la Figura 4.5 se observa la aplicación, en la cual se determina las librerías a utilizar y las variables usadas. Las variables son subrutinas que facilitan la programación del microcontrolador. Se incluye la librería de comunicación serial y la librería para el módulo GLCD.

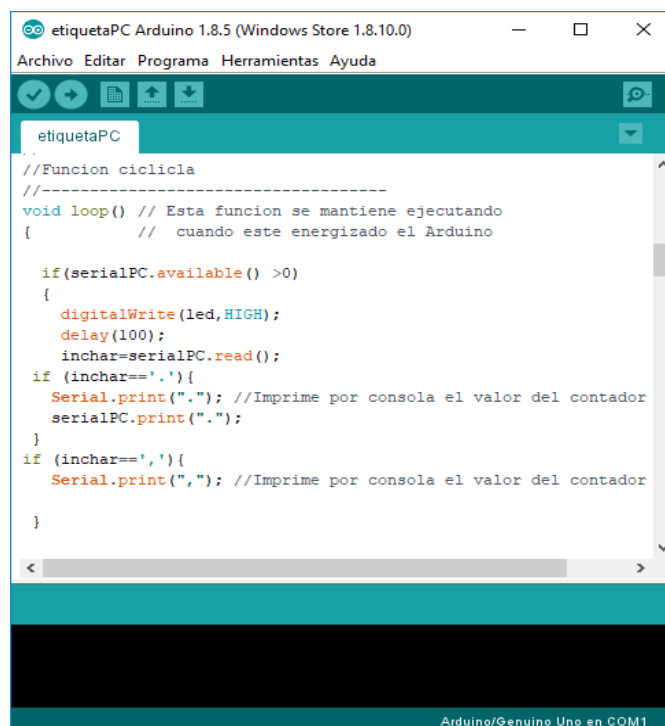


```
etiquetaPC
//-----
#include <SoftwareSerial.h>//incluimos SoftwareSerial
SoftwareSerial serialPC(2, 3);//Declaramos el pin 2 rx y 3 tx

int led = 13;           //Pin donde se encuentra el LED, salida
char leer;             //Variable donde se almacena la letra
boolean prendido=false; //Estado LED la primera vez, apagado
int contador = 0;      //Variable del contador
char inchar;
int dato;
//-----
//Funcion principal
//-----
void setup() // Se ejecuta cada vez que el Arduino se inicia
{
  Serial.begin(9600); //Inicia comunicación serial
  pinMode(led, OUTPUT); //Configurar el LED como una salida
  serialPC.begin(9600);
}
}
Arduino/Genuino Uno en COM1
```

Figura 4.5: Librerías y variables de la controladora principal.
Fuente: (Elaborado por el autor).

En la figura 4.6 es el desarrollo del programa para la controladora principal, en la cual mediante el puerto serial envía las letras que serán presentadas en nuestras etiquetas electrónicas. Se utilizó códigos ASCII para diferenciar el ingreso de las letras entre una etiqueta a otra.



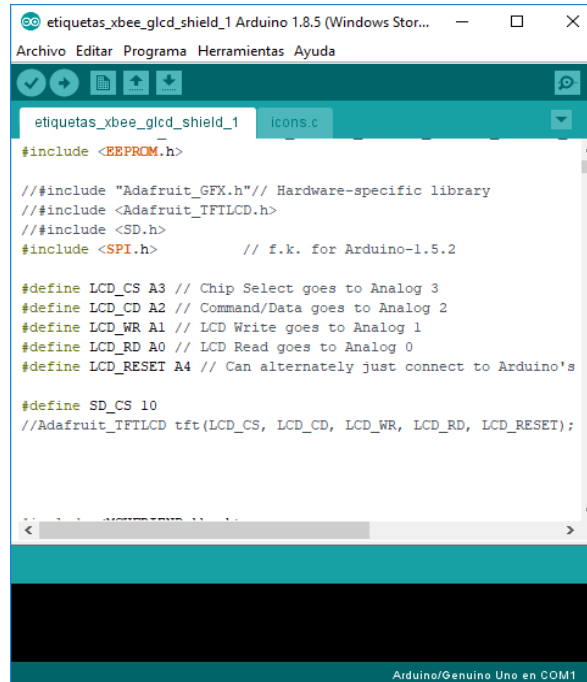
```
etiquetaPC
//Funcion ciclicla
//-----
void loop() // Esta funcion se mantiene ejecutando
{           // cuando este energizado el Arduino

  if(serialPC.available() >0)
  {
    digitalWrite(led,HIGH);
    delay(100);
    inchar=serialPC.read();
  if (inchar=='.'){
    Serial.print("."); //Imprime por consola el valor del contador
    serialPC.print(".");
  }
  if (inchar==','){
    Serial.print(","); //Imprime por consola el valor del contador
  }
}
}
Arduino/Genuino Uno en COM1
```

Figura 4.6: Programa para envío de letras controladora principal.
Fuente: (Elaborado por el autor)

4.2.5 Programación de las etiquetas electrónicas.

En la Figura 4.7 se muestran las librerías que ayudan a controlar la memoria EEPROM, el display GLCD y el puerto serial.



```
etiquetas_xbee_glcd_shield_1 icons.c
#include <EEPROM.h>

// #include "Adafruit_GFX.h" // Hardware-specific library
// #include <Adafruit_TFTLCD.h>
// #include <SD.h>
#include <SPI.h> // f.k. for Arduino-1.5.2

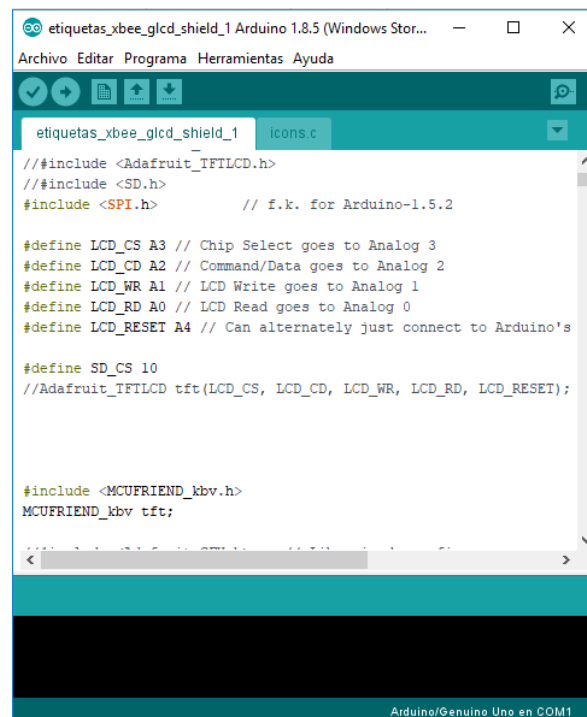
#define LCD_CS A3 // Chip Select goes to Analog 3
#define LCD_CD A2 // Command/Data goes to Analog 2
#define LCD_WR A1 // LCD Write goes to Analog 1
#define LCD_RD A0 // LCD Read goes to Analog 0
#define LCD_RESET A4 // Can alternately just connect to Arduino's

#define SD_CS 10
//Adafruit_TFTLCD tft(LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);
```

Figura 4.7: Librería de las etiquetas electrónicas.

Fuente: (Elaborado por el autor).

En la Figura 4.8 se define las variables para cambiar la descripción del producto y precio en la GLCD.



```
etiquetas_xbee_glcd_shield_1 icons.c
// #include <Adafruit_TFTLCD.h>
// #include <SD.h>
#include <SPI.h> // f.k. for Arduino-1.5.2

#define LCD_CS A3 // Chip Select goes to Analog 3
#define LCD_CD A2 // Command/Data goes to Analog 2
#define LCD_WR A1 // LCD Write goes to Analog 1
#define LCD_RD A0 // LCD Read goes to Analog 0
#define LCD_RESET A4 // Can alternately just connect to Arduino's

#define SD_CS 10
//Adafruit_TFTLCD tft(LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);

#include <MCFRIEND_kbv.h>
MCFRIEND_kbv tft;
```

Figura 4.8: Variables LCD etiquetas electrónicas.

Fuente: (Elaborado por el autor).

En la Figura 4.9 se tiene la comunicación serial y las secuencias para guardar las letras en la memoria EEPROM y no se pierda los datos.

```

etiquetas_xbee_glcd_shield_1 icons.c
Archivo Editar Programa Herramientas Ayuda

if (!Serial) delay(5000); //allow some time for Leor
Serial.println("Serial took " + String((millis() - when)) + "n
// tft.reset(); //hardware reset
uint16_t ID = tft.readID(); //
Serial.print("ID = 0x");
Serial.println(ID, HEX);
if (ID == 0xD3D3) ID = 0x9481; // write-only shield
// ID = 0x9329; // force ID
tft.begin(ID);

letra0x = EEPROM.read(0);
letrax=letra0x;
convertir();
letra0=valor;

letralx = EEPROM.read(1);

```

Figura 4.9: Recepción y almacenamiento de letras en la memoria EEPROM.
Fuente: (Elaborado por el autor)

4.2.6 Programación de la aplicación para el usuario.

La aplicación se la realizó en el programa Visual Basic 6.0 que es un programa que trabaja en todas las versiones de Windows. En el programa Visual figura 4.10 se generan los botones que serán modificados, y se les asigna un código ASCII que será el que determine el cambio de variables en las etiquetas electrónicas.

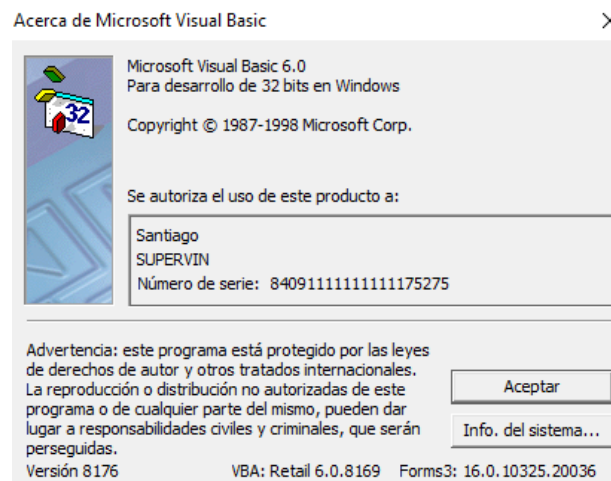


Figura 4.10: Programa Visual Basic utilizado.
Fuente: (Elaborado por el autor).

En la figura 4.11 se indica una muestra de la programación en Visual Basic en la cual se asigna un código ASCII a cada botón de la aplicación, para que sean enviados valores a la controladora principal.

```
Private Sub Command7_Click()  
MSComml.Output = "e"  
End Sub  
  
Private Sub Command8_Click()  
MSComml.Output = "r"  
End Sub  
  
Private Sub Command9_Click()  
MSComml.Output = "t"  
End Sub  
  
Private Sub Form_Load()  
MSComml.PortOpen = True  
Timer1.Interval = 1  
End Sub  
  
Private Sub Command1_Click()  
MSComml.Output = "."  
End Sub
```

Figura 4.11: Programación en Visual Basic 6.0.

Fuente: (Elaborado por el autor).

En la figura 4.12 se muestra la aplicación que será utilizada por el usuario. En ella por medio de botones se tiene las opciones de modificar: precio, descripción, precio de oferta de acuerdo con las necesidades del usuario.

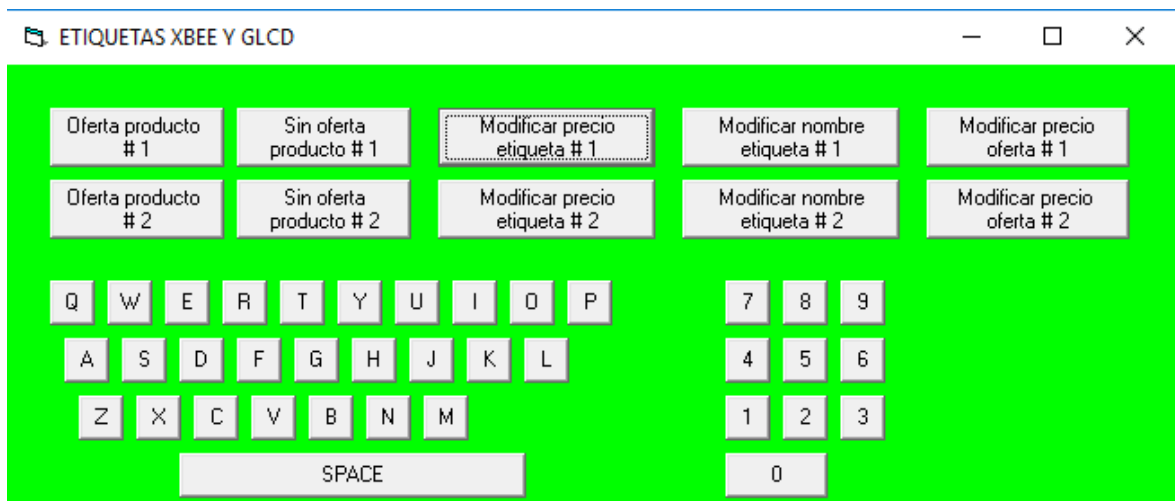


Figura 4.12: Presentación de la aplicación para el usuario.

Fuente: (Elaborado por el autor).

4.3 Pruebas de funcionamiento.

Para verificar el funcionamiento de las etiquetas electrónicas, se realizaron pruebas por medio de alcance de transmisión en metros, con y sin obstáculos. Estas pruebas se detallan en la tabla 4.1

Tabla 4.1: Pruebas de alcance de etiquetas electrónicas.

Etiqueta	Distancia	Obstáculo tela	Sin obstáculo	Respuesta
Etiqueta 1	5m	x		Transmite
Etiqueta 2	5m		x	Transmite
Etiqueta 1	20m	x		Transmite
Etiqueta 2	20m		x	Transmite
Etiqueta 1	50m	x		Transmite
Etiqueta 2	50m		x	Transmite
Etiqueta 1	75m	x		Pierde datos
Etiqueta 2	75m		x	Transmite
Etiqueta 1	90m	x		No transmite
Etiqueta 2	90m		x	Transmite
Etiqueta 1	100m	x		No transmite
Etiqueta 2	100m		x	No transmite

Fuente:(Elaborado por el autor)

También se realizaron pruebas de pérdida de datos al quitar la alimentación de voltaje en las etiquetas electrónicas.

Tabla 4.2: Pruebas de pérdida de datos sin voltaje.

Etiqueta	Sin voltaje	Valores se mantienen
Etiqueta 1	x	si
Etiqueta 2	x	si

Fuente:(Elaborado por el autor)

Adicional se realizaron pruebas de duración de la batería con las etiquetas y del tiempo de carga.

Los resultados se muestran en la tabla 4.3.

Tabla 4.3: Pruebas de batería en etiquetas electrónicas.

TIEMPO DE DESCARGA			
ETIQUETA:	Precio normal	Precio de oferta	TIEMPO DE CARGA
1	13 horas	12 horas	2 horas
2	12:45 horas	12:05 horas	2 horas

Fuente:(Elaborado por el autor).

4.4 Análisis de resultados.

Los obstáculos utilizados en la tabla 4.1 son los que se encuentran en un Supermercado

una combinación de tela con plásticos. En base a los resultados obtenidos en las tablas 4.1 se determina que el alcance de los módulos Zigbee está sujeto a la distancia a la que se encuentra el controlador principal y a los obstáculos presentes. Por ello hay que tener presente que para realizar correctamente el cambio de precios y la descripción de nuestra etiqueta electrónica debemos estar por lo menos a 50m.

De acuerdo con la tabla 4.2 no se tiene problemas de pérdida de datos en las etiquetas por cortes de voltaje. El tiempo de duración de la batería es de 12 horas en un producto en oferta, un Supermercado se encuentra abierto 12 horas al día, por lo que las etiquetas se mantendrán trabajando el tiempo que se necesita, para luego ser cargadas en la noche. Las etiquetas cuentan con un circuito de carga por medio de un conector USB, con ello se logra que la etiqueta sea sellada y se cargue externamente.

5 CONCLUSIONES Y RECOMENDACIONES.

5.1 CONCLUSIONES.

Al finalizar el proyecto se tiene las siguientes conclusiones:

- Se determinó que el microcontrolador ATM 328P es la mejor opción para implementar el proyecto, por la cantidad de memoria y facilidad de programación. También parámetros como 2 voltajes necesarios para su funcionamiento.
- A realizar los circuitos de comunicación y Visualización de las etiquetas electrónicas, se debe tener consideraciones como: la protección al microcontrolador y al módulo Zigbee por cambio de voltaje o señales con picos o atenuación.
- La aplicación para controlar las etiquetas electrónicas se realizó con el programa Visual Basic, porque es fácil de programación y se tiene la opción de utilizar botones gráficos amigables para el usuario.
- Al realizar la implementación de la etiqueta de acuerdo al diseño, se tuvo que realizar unos cambios adicionales como un regulador LM117 que se le colocó en la entrada de la controladora principal en el módulo USB-TTL.
- Al realizar las pruebas de funcionamiento, en general la etiqueta electrónica cumple con las necesidades de un Supermercado como el tiempo de funcionamiento y el alcance requerido para el cambio de información en las mismas.
- Al utilizar módulos Zigbee debemos tener presente si los vamos a utilizar en modo punto a punto o modo punto multipunto, para configurarlos correctamente.
- Zigbee es un sistema que permite un alto número de escalabilidad, permitiendo hasta un total de 65534 módulos interconectarse, por ello podríamos utilizar todas las etiquetas que necesitemos.
- El campo de acción de la tecnología Zigbee frente a otras tecnologías, está definido y no tiene competidores.

5.2 RECOMENDACIONES.

- Al instalar la aplicación para el control de las etiquetas, hay que tener presente el Sistema operativo del PC, ya que los drivers difieren dependiendo del Sistema operativo Windows XP, 7 y 10.
- En las etiquetas electrónicas se debe habilitar el reset ante cualquier falla de software.
- Se debe utilizar la memoria EEPROM de los ATM 328P en las etiquetas electrónicas, por si existe algún corte de voltaje, con ello no se pierde la información en las etiquetas.
- Al realizar la alimentación de voltaje por medio de baterías, hay que tener en cuenta que se necesita 2 voltajes, el módulo Zigbee 3.3V y el display GLCD 5V.
- El alcance de la controladora varía de 30 a 90m aproximado ya que los obstáculos disminuyen el alcance de las ondas electromagnéticas por lo que es aconsejable manejar una distancia intermedia para no tener problemas de comunicación.
- La tecnología Zigbee puede ser vulnerable ante ataques, por lo que se debe utilizar, para sus propósitos definidos.

6 REFERENCIAS BIBLIOGRÁFICAS.

- Atmel. (1 de 11 de 2018). *Atmel*. Obtenido de http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Summary.pdf
- carlos@nergiza.com. (26 de 11 de 2014). *NERGIZA*. Recuperado el 17 de 8 de 2018, de <https://nergiza.com/como-hacer-tu-propio-powerbank-a-medida/>
- Castillo, J. (23 de 02 de 2018). *Repositorio Universidad Israel*. Recuperado el 15 de 06 de 2018, de <http://repositorio.uisrael.edu.ec/bitstream/47000/1544/1/UISRAEL-EC-ELDT-378.242-2018-002.pdf>
- Domodesk S.L. (24 de 07 de 2018). *Domodesk*. Obtenido de <https://www.domodesk.com/216-a-fondo-zigbee.html>
- Evans, B. W. (2011). *arduino programming notebook*. California: Ardumania.
- Farahani, S. (2008). *ZigBee Wireless Networks and Transceive*. Oxford: ELSEVIER.
- Gutierrez, M. J. (10 de 08 de 2015). *El Español*. Recuperado el 10 de 8 de 2018, de <https://elandroidelibre.elespanol.com/2015/08/todo-sobre-zigbee-la-tecnologia-ultrabarata-para-comunicacion-inalambrica.html>
- Heather Chesterman. (19 de 04 de 2018). *Zigbee Aliance*. Obtenido de <https://www.zigbee.org/zigbee-for-developers/zigbee-3-0/>
- Hernandez, J. C. (2015). Protocolo de comunicaciones inalámbrico. *2015 Workshop on Engineering Applications* (págs. 1-12). Bogota: Systems Ecastats.
- Joe. (27 de 07 de 2017). *Groups.io*. Recuperado el 07 de 06 de 2018, de <https://groups.io/g/BITX20/message/30602>
- Kinney, P. (2003). Wireless Control that Simply. *Communications Design Conference* (págs. 2-20). Washington, D.C: Kinney Consulting LLC.
- Nasir, S. Z. (5 de 1 de 2016). *XBee Library for Proteus*. Obtenido de <http://www.theengineeringprojects.com/2016/01/xbee-library-proteus.html>
- Ojeda, L. T. (s.f.). *Xbee.cl*. Obtenido de <http://xbee.cl/>
- Paz, J. S. (1 de 08 de 2018). *vdocuments.site*. Recuperado el 10 de 07 de 2018, de <https://vdocuments.site/documents/transmision-datos-zigbeepdf.html>
- Sebastian, C. (5 de 08 de 2018). *Electgpl*. Obtenido de <http://electgpl.blogspot.com/2016/06/el-atmega328p.html>

stiboons. (30 de 01 de 2017). *Kerbal*. Recuperado el 01 de 06 de 2018, de <https://forum.kerbalspaceprogram.com/index.php?/topic/155796-ksp-144-kerbal-simpit-a-new-ksp-serial-mod-for-hardware-controllers-128/&page=6&tab=comments#comment-3106345>

Varios. (2016). *Zigbee Network Protocols and Applications*. New York: CRC Press.

verdelj. (14 de 04 de 2013). *Instructables*. Obtenido de <https://www.instructables.com/id/Usb-to-SerialTTL-adapter/>


7 ANEXOS:

Anexo 1: Precios de elementos electrónicos usados.



Modulo Lcd Tft Para Arduino Uno O Mega

U\$S 15

 Envío a todo el país

10 vendidos - Guayas

AV
Electronics



Xbee Usb Adapter

U\$S 10

6 vendidos - Pichincha (Quito)

Xbee USB Adapter

www.avelectronics.cc



Xbee S2c

U\$S 48

Imbabura

Anexo 2: Xbee en el mercado.

Xbee	Max data rate	Frecuency band	Transmit power	Antena	IO pins digital	ADC inputs	Range
 <p>XBee 1mW PCB Antenna</p>	115.2 kbP	2.4 GHz	1 mW (+0 dBm)	Built-in	8	(7) 10-bit	300ft (100m)
 <p>XBee 1mW Wire Antenna</p>	250kb ps	2.4 GHz	1mW output (+0dBm)	Built-in	8	(6) 10-bit	300ft (100m)
 <p>XBee 2mW Chip Antenna – Series 2</p>	250kb ps	2.4 GHz	2mW output (+3dBm)	Built-in	8	(6) 10-bit	400ft (120m)

 <p>XBee 2mW PCB Antenna – Series 2 (Zigbee Mesh)</p>	250kps	2.4 GHz	2mW output (+3dBm).	Built-in	8	(6) 10-bit	400ft (120m)
 <p>XBee 2mW RPSMA – Series 2</p>	250kps	2.4 GHz	2mW output (+3dBm)	RPSMA	8	(6) 10-bit	400ft (120m)
 <p>XBee 2mW Wire Antenna – Series 2 (Mesh)</p>	250kps	2.4 GHz	2mW output (+3dBm)	Built-in	8	(6) 10-bit	400ft (120m)
 <p>XBee Pro 50mW</p>	250kps	2.4 GHz	50mW output (+17dBm).	RPSMA	8	(6) 10-bit	1 mile (1600m)

Serie 2.5 Wire Antena							
 XBee Pro 60mW serie 1 PCB Antena	250kps	2.4 GHz	50mW output (+17dBm).	Built-in	8	(4) 10-bit	1 mile (1600 m)
 XBee Pro 60mW Wire Antenna	250kps	2.4 GHz	60mW output (+18dBm)	Built-in	8	(6) 10-bit	1 mile (1600 m)
 XBee Pro 900 RPSMA	156 Kbps	ISM de 900MHz	50 mW (+17 dBm)	RPSMA	10	(6) 10-bit	6 miles (10 km)
 XBee Pro 900 XSC RPSMA	9.6kps	ISM de 900MHz	100 mW power output	RPSMA	none	none	15 miles

Fuente: (Ojeda, s.f.)

Anexo 3: Data sheet AT MEGA 328P



Device Overview Summary

The high-performance Microchip picoPower 8-bit AVR RISC-based microcontroller combines 32KB ISP flash memory with read-while-write capabilities, 1024B EEPROM, 2KB SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible timer/counters with compare modes, internal and external interrupts, serial programmable USART, a byte-oriented 2-wire serial interface, SPI serial port, a 6-channel 10-bit A/D converter (8-channels in TQFP and QFN/MLF packages), programmable watchdog timer with internal oscillator, and five software selectable power saving modes. The device operates between 1.8-5.5 volts.

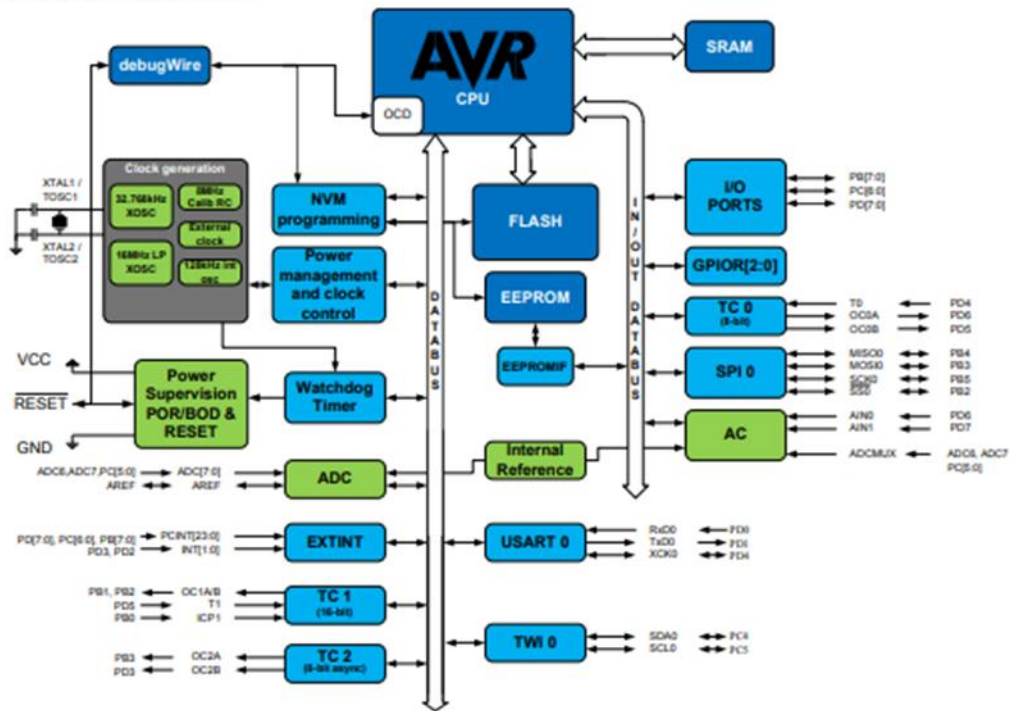
By executing powerful instructions in a single clock cycle, the device achieves throughputs approaching 1 MIPS per MHz, balancing power consumption and processing speed.

Parametrics

Name	Value
Program Memory Type	Flash
Program Memory Size (KB)	32
CPU Speed (MIPS/DMIPS)	20
SRAM Bytes	2,048
Data EEPROM/HEF (bytes)	1024
Digital Communication Peripherals	1-UART, 2-SPI, 1-I2C
Capture/Compare/PWM Peripherals	1 Input Capture, 1 CCP, 6PWM
Timers	2 x 8-bit, 1 x 16-bit
Number of Comparators	1
Temperature Range (C)	-40 to 85
Operating Voltage Range (V)	1.8 to 5.5
Pin Count	32
Low Power	Yes

Block Diagram

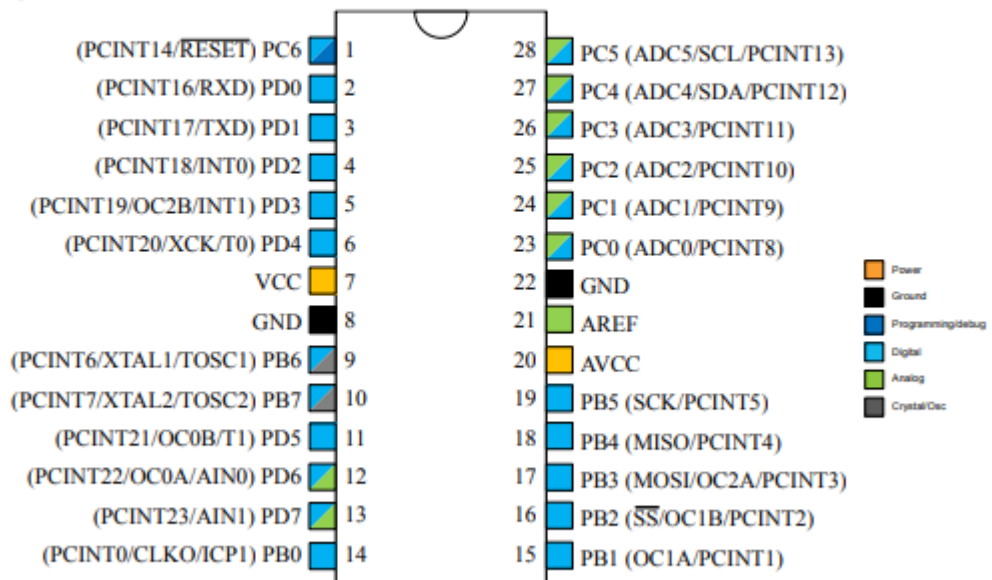
Figure 4-1. Block Diagram



5. Pin Configurations

5.1. Pin-out

Figure 5-1. 28-pin PDIP



5.2. Pin Descriptions

5.2.1. VCC Digital supply voltage.

5.2.2. GND Ground.

5.2.3. Port B (PB[7:0]) XTAL1/XTAL2/TOSC1/TOSC2 Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive

characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit. Atmel ATmega328/P [DATASHEET] Atmel-42735B-328/P_Datasheet_Summary-11/2016 12 Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier. If the Internal Calibrated RC Oscillator is used as chip clock source, PB[7:6] is used as TOSC[2:1] input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

5.2.4. Port C (PC[5:0]) Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC[5:0] output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

5.2.5. PC6/RESET If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C. If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a Reset. The various special features of Port C are elaborated in the Alternate Functions of Port C section.

5.2.6. Port D (PD[7:0]) Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

5.2.7. AVCC is the supply voltage pin for the A/D Converter, PC[3:0], and PE[3:2]. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter. Note that PC[6:4] use digital supply voltage, VCC.

5.2.8. AREF is the analog reference pin for the A/D Converter.

5.2.9. ADC[7:6] (TQFP and VFQFN Package Only) In the TQFP and VFQFN package.

Anexo 4: Programación etiqueta electrónica en Arduino IDE.

```
#include <SPI.h>          // f.k. for Arduino-1.5.2

#define LCD_CS A3 // Chip Select goes to Analog 3

#define LCD_CD A2 // Command/Data goes to Analog 2

#define LCD_WR A1 // LCD Write goes to Analog 1

#define LCD_RD A0 // LCD Read goes to Analog 0

#define LCD_RESET A4 // Can alternately just connect to Arduino's reset pin

#define SD_CS 10

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

#include <MCUFRIEND_kbv.h>

MCUFRIEND_kbv tft;

//#include <Adafruit_GFX.h> // Libreria de graficos

// Assign human-readable names to some common 16-bit color values:

#define BLACK 0x0000
```

```

#define BLUE 0x001F
#define RED 0xF800
#define GREEN 0x07E0
#define CYAN 0x07FF
#define MAGENTA 0xF81F
#define YELLOW 0xFFE0
#define WHITE 0xFFFF

#ifndef min
#define min(a, b) (((a) < (b)) ? (a) : (b))
#endif
int bandera=0; int digito3;
char leer; int digito5;
char inchar; int digito4;
int digito2; char letra12; int letrax2;
int digito1; char letra13;
int digito5; char letra14; int letra0x;
int digito4; char letra15; int letra1x;
int digito3; char letra16; int letra2x;
int digito2; char letra17; int letra3x;
int digito1; char letra18; int letra4x;
int contadorprecio; char letra19; int letra5x;
char letra0; char letra20; int letra6x;
char letra1; char letra21; int letra7x;
char letra2; char letra22; int letra8x;
char letra3; char letra23; int letra9x;
char letra4; char letra24; int letra10x;
char letra5; char letra25; int letra11x;
char letra6; char letra26; int letra12x;
char letra7; int letra13x;
char letra8; char valor; int letra14x;
char letra9; int letrax; int letra15x;
char letra10; int letra16x;
char letra11; char valor2; int letra17x;

```

```

int letra18x;                int letra20x;                int letra22x;
int letra19x;                int letra21x;                int letra23x;
int letra24x;                int letra26x;                int oferta;
int letra25x;

void setup(void);
void loop(void);
unsigned long testFillScreen();
unsigned long testText();
unsigned long testLines(uint16_t color);
unsigned long testFastLines(uint16_t color1, uint16_t color2);
unsigned long testRects(uint16_t color);
unsigned long testFilledRects(uint16_t color1, uint16_t color2);
unsigned long testFilledCircles(uint8_t radius, uint16_t color);
unsigned long testCircles(uint8_t radius, uint16_t color);
unsigned long testTriangles();
unsigned long testFilledTriangles();
unsigned long testRoundRects();
unsigned long testFilledRoundRects();
void progmemPrint(const char *str);
void progmemPrintln(const char *str);

void runtests(void);

uint16_t g_identifier;

extern const uint8_t hanzi[];
void showhanzi(unsigned int x, unsigned int y, unsigned char index)
{
    uint8_t i, j, c, first = 1;
    uint8_t *temp = (uint8_t*)hanzi;
    uint16_t color;

    tft.setAddrWindow(x, y, x + 31, y + 31); //设置区域

```

```

temp += index * 128;
for (j = 0; j < 128; j++)
{
    c = pgm_read_byte(temp);
    for (i = 0; i < 8; i++)
    {
        if ((c & (1 << i)) != 0)
        {
            color = RED;
        }
        else
        {
            color = BLACK;
        }
        tft.pushColors(&color, 1, first);
        first = 0;
    }
    temp++;
}

void setup(void) {
    Serial.begin(9600);
    uint32_t when = millis();
    // while (!Serial); //hangs a Leonardo until you connect a Serial
    if (!Serial) delay(5000); //allow some time for Leonardo
    Serial.println("Serial took " + String((millis() - when)) + "ms to start");
    // tft.reset(); //hardware reset
    uint16_t ID = tft.readID(); //
    Serial.print("ID = 0x");
    Serial.println(ID, HEX);
    if (ID == 0xD3D3) ID = 0x9481; // write-only shield
    // ID = 0x9329; // force ID
}

```

```
tft.begin(ID);  
letra0x = EEPROM.read(0);  
letrax=letra0x;  
convertir();  
letra0=valor;
```

```
letra1x = EEPROM.read(1);  
letrax=letra1x;  
convertir();  
letra1=valor;
```

```
letra2x = EEPROM.read(2);  
letrax=letra2x;  
convertir();  
letra2=valor;
```

```
letra3x = EEPROM.read(3);  
letrax=letra3x;  
convertir();  
letra3=valor;
```

```
letra4x = EEPROM.read(4);  
letrax=letra4x;  
convertir();  
letra4=valor;
```

```
letra5x = EEPROM.read(5);  
letrax=letra5x;  
convertir();  
letra5=valor;
```

```
letra6x = EEPROM.read(6);
```

```
letrax=letra6x;  
convertir();  
letra6=valor;
```

```
letra7x = EEPROM.read(7);  
letrax=letra7x;  
convertir();  
letra7=valor;
```

```
letra8x = EEPROM.read(8);  
letrax=letra8x;
```

```

convertir();
letra8=valor;

letra9x = EEPROM.read(9);
letrax=letra9x;
convertir();
letra9=valor;

letra10x = EEPROM.read(10);
letrax=letra10x;
convertir();
letra10=valor;

letra11x = EEPROM.read(11);
letrax=letra11x;
convertir();
letra11=valor;

letra12x = EEPROM.read(12);
letrax=letra12x;
convertir();
letra12=valor;

letra13x = EEPROM.read(13);
letrax=letra13x;
convertir();
letra13=valor;

letra14x = EEPROM.read(14);
letrax=letra14x;
convertir();
letra14=valor;

letra15x = EEPROM.read(15);
letrax=letra15x;
convertir();
letra15=valor;

letra16x = EEPROM.read(16);
letrax=letra16x;
convertir();
letra16=valor;

letra17x = EEPROM.read(17);
letrax=letra17x;
convertir();
letra17=valor;

letra18x = EEPROM.read(18);
letrax=letra18x;
convertir();
letra18=valor;

letra19x = EEPROM.read(19);
letrax=letra19x;
convertir();
letra19=valor;

letra20x = EEPROM.read(20);
letrax=letra20x;
convertir();
letra20=valor;

letra21x = EEPROM.read(21);
letrax=letra21x;

```

```

convertir();
letra21=valor;

letra22x = EEPROM.read(22);
letrax=letra22x;
convertir();
letra22=valor;

letra24=valor;

letra25x = EEPROM.read(25);
letrax=letra25x;
convertir();
letra25=valor;

letra26x = EEPROM.read(26);
letrax=letra26x;
convertir();
letra26=valor;

digito5 = EEPROM.read(50);
digito4 = EEPROM.read(51);
digito3 = EEPROM.read(52);
digito2 = EEPROM.read(53);
digito1 = EEPROM.read(54);

oferta = EEPROM.read(60);

digitob5 = EEPROM.read(70);
digitob4 = EEPROM.read(71);
digitob3 = EEPROM.read(72);
digitob2 = EEPROM.read(73);

letra23x = EEPROM.read(23);
letrax=letra23x;
convertir();
letra23=valor;

letra24x = EEPROM.read(24);
letrax=letra24x;
convertir();

```

```

digitob1 = EEPROM.read(74);

/*

tft.reset();
tft.begin(0x9341);
if (!SD.begin(SD_CS))
{
    Serial.println(F("ERROR!"));
    return;
}
Serial.println(F("OK!"));

tft.setRotation(0);
bmpDraw("1.bmp", 0, 0);
delay(1000);

*/

uint8_t aspect;
uint16_t pixel;
const char *aspectname[] = {
    "PORTRAIT", "LANDSCAPE", "PORTRAIT_REV", "LANDSCAPE_REV"
};
const char *colormname[] = { "BLUE", "GREEN", "RED", "GRAY" };
uint16_t colormask[] = { 0x001F, 0x07E0, 0xF800, 0xFFFF };
uint16_t dx, rgb, n, wid, ht, msglin;
tft.setRotation(1);//establece vertical u horizontal
//runtests(); //cuadrados que se desplazan de un lado a otro

}

#ifdef MCUFRIEND_KBV_H_
uint16_t scrollbuf[320]; // my biggest screen is 320x480

```



```

#define READGRAM(x, y, buf, w, h) tft.readGRAM(x, y, buf, w, h)

#else

uint16_t scrollbuf[320]; // Adafruit only does 240x320

// Adafruit can read a block by one pixel at a time
int16_t READGRAM(int16_t x, int16_t y, uint16_t *block, int16_t w, int16_t h)
{
    uint16_t *p;
    for (int row = 0; row < h; row++) {
        p = block + row * w;
        for (int col = 0; col < w; col++) {
            *p++ = tft.readPixel(x + col, y + row);
        }
    }
}

#endif

void windowScroll(int16_t x, int16_t y, int16_t wid, int16_t ht, int16_t dx, int16_t dy, uint16_t *buf)
{
    if (dx) for (int16_t row = 0; row < ht; row++) {
        READGRAM(x, y + row, buf, wid, 1);
        tft.setAddrWindow(x, y + row, x + wid - 1, y + row);
        tft.pushColors(buf + dx, wid - dx, 1);
        tft.pushColors(buf + 0, dx, 0);
    }
    if (dy) for (int16_t col = 0; col < wid; col++) {
        READGRAM(x + col, y, buf, 1, ht);
        tft.setAddrWindow(x + col, y, x + col, y + ht - 1);
        tft.pushColors(buf + dy, ht - dy, 1);
        tft.pushColors(buf + 0, dy, 0);
    }
}

void printmsg(int row, const char *msg)

```

```

{
  tft.setTextColor(YELLOW, BLACK);
  tft.setCursor(0, row);
  tft.println(msg);
}

```

```

void loop(void) {

```

```

  if (bandera==0){

```

```

    tft.fillScreen(GREEN);

```

```

    tft.setCursor(5, 5);

```

```

    tft.setTextColor(BLACK);

```

```

    tft.setTextSize(4);

```

```

    tft.println("SUPERMERCADO");

```

```

    tft.setCursor(55, 40);

```

```

    tft.setTextSize(2);

```

```

    tft.setTextColor(BLACK);

```

```

    tft.println("TODO A
MENOR PRECIO");

```

```

    tft.setCursor(110, 60);

```

```

    tft.setTextSize(1);

```

```

    tft.setTextColor(BLACK);

```

```

    tft.println("Ubicacion :
A1");

```

```

    tft.setCursor(5,75);

```

```

    tft.setTextSize(3);

```

```

    tft.setTextColor(BLUE);

```

```

    tft.print(letra0);

```

```

    tft.setCursor(25,75);

```

```

    tft.setTextSize(3);

```

```

    tft.setTextColor(BLUE);

```

```

    tft.print(letra1);

```

```

    tft.setCursor(45,75);

```

```

    tft.setTextSize(3);

```

```

    tft.setTextColor(BLUE);

```

```

    tft.print(letra2);

```

```

    tft.setCursor(65,75);

```

```

    tft.setTextSize(3);

```

```

    tft.setTextColor(BLUE);

```

```

    tft.print(letra3);

```

```

    tft.setCursor(85,75);

```

```

    tft.setTextSize(3);

```

```

    tft.setTextColor(BLUE);

```

```

    tft.print(letra4);

```

```

    tft.setCursor(105,75);

```

```

    tft.setTextSize(3);

```

```

    tft.setTextColor(BLUE);

```

```

    tft.print(letra5);

```

```

    tft.setCursor(125,75);

```

```

    tft.setTextSize(3);

```

```

    tft.setTextColor(BLUE);

```

```

    tft.print(letra6);

```

```

    tft.setCursor(145,75);

```

```

    tft.setTextSize(3);

```

```

    tft.setTextColor(BLUE);

```

tft.print(letra7);	tft.print(letra9);	tft.print(letra11);
tft.setCursor(165,75);	tft.setCursor(205,75);	tft.setCursor(245,75);
tft.setTextSize(3);	tft.setTextSize(3);	tft.setTextSize(3);
tft.setTextColor(BLUE);	tft.setTextColor(BLUE);	tft.setTextColor(BLUE);
tft.print(letra8);	tft.print(letra10);	tft.println(letra12);
tft.setCursor(185,75);	tft.setCursor(225,75);	tft.setCursor(265,75);
tft.setTextSize(3);	tft.setTextSize(3);	
tft.setTextColor(BLUE);	tft.setTextColor(BLUE);	
tft.setTextSize(3);	tft.println(letra17);	tft.setCursor(165,100);
tft.setTextColor(BLUE);		tft.setTextSize(3);
tft.println(letra13);	tft.setCursor(85,100);	tft.setTextColor(BLUE);
	tft.setTextSize(3);	tft.println(letra22);
tft.setCursor(5,100);	tft.setTextColor(BLUE);	
tft.setTextSize(3);	tft.println(letra18);	tft.setCursor(185,100);
tft.setTextColor(BLUE);		tft.setTextSize(3);
tft.println(letra14);	tft.setCursor(105,100);	tft.setTextColor(BLUE);
	tft.setTextSize(3);	tft.println(letra23);
tft.setCursor(25,100);	tft.setTextColor(BLUE);	
tft.setTextSize(3);	tft.println(letra19);	tft.setCursor(205,100);
tft.setTextColor(BLUE);		tft.setTextSize(3);
tft.println(letra15);	tft.setCursor(125,100);	tft.setTextColor(BLUE);
	tft.setTextSize(3);	tft.println(letra24);
tft.setCursor(45,100);	tft.setTextColor(BLUE);	
tft.setTextSize(3);	tft.println(letra20);	tft.setCursor(225,100);
tft.setTextColor(BLUE);		tft.setTextSize(3);
tft.println(letra16);	tft.setCursor(145,100);	tft.setTextColor(BLUE);
	tft.setTextSize(3);	tft.println(letra25);
tft.setCursor(65,100);	tft.setTextColor(BLUE);	
tft.setTextSize(3);	tft.println(letra21);	//tft.setCursor(245,100);
tft.setTextColor(BLUE);		//tft.setTextSize(3);

```

//tft.setTextColor(BLUE);

//tft.println(letra26);
tft.setTextSize(3);

tft.setCursor(22, 170);
tft.setTextSize(8);
tft.setTextColor(BLACK);
tft.print(digito5);
tft.print(digito4);
tft.print(digito3);
tft.print(".");
tft.print(digito2);
tft.print(digito1);

bandera=1;
}

if(bandera==1){
  leer=Serial.read();
  if (leer == '.') {
    bandera = 2;
    delay(100);
    contadorprecio=5;
  }

  if (leer == ','){
    bandera = 3;
  }

  tft.setCursor(5, 130);
  tft.setTextColor(BLACK);

  delay(100);
  contadorprecio=26;
}

if (leer == '+'){
  oferta=0;
  EEPROM.write(60,oferta);
  bandera = 0;
  delay(100);
}

if (leer == '-'){
  oferta=1;
  EEPROM.write(60,oferta);
  bandera = 0;
  delay(100);
}

}

if (leer == '/') {
  bandera = 4;
  delay(100);
  contadorprecio=5;
}

}

delay(2000);
bandera=0;
}

}

if (bandera == 2){
  if (contadorprecio==0){

```

```

bandera=0;
}
if(Serial.available(>0){
  leer=Serial.read();
  if (leer== '0'){
    inchar=0;
  }
  if (leer== '1'){
    inchar=1;
  }
  if (leer== '2'){
    inchar=2;
  }
  if (leer== '3'){
    inchar=3;
  }
  if (leer== '4'){
    inchar=4;
  }
  if (leer== '5'){
    inchar=5;
  }
  if (leer== '6'){
    inchar=6;
  }
  if (leer== '7'){
    inchar=7;
  }
  if (leer== '8'){
    inchar=8;
  }
  if (leer== '9'){
    inchar=9;
  }
}
if (contadorprecio==1){
  digito1=inchar;
  EEPROM.write(54,digito1);
  contadorprecio --;
}
if (contadorprecio==2){
  digito2=inchar;
  EEPROM.write(53,digito2);
  contadorprecio --;
}
if (contadorprecio==3){
  digito3=inchar;
  EEPROM.write(52,digito3);
  contadorprecio --;
}
if (contadorprecio==4){
  digito4=inchar;
  EEPROM.write(51,digito4);
  contadorprecio --;
}
if (contadorprecio==5){
  digito5=inchar;
  EEPROM.write(50,digito5);
  contadorprecio --;
}
}
if (bandera == 3){
  if (contadorprecio==0){
    bandera=0;
  }
  if(Serial.available(>0){
    leer=Serial.read();
    if (leer== ' '){
      inchar=' ';
    }
    if (leer== 'a'){
      inchar='a';
    }
    if (leer== 'b'){
      inchar='b';
    }
    if (leer== 'c'){
      inchar='c';
    }
    if (leer== 'd'){
      inchar='d';
    }
    if (leer== 'e'){
      inchar='e';
    }
    if (leer== 'f'){
      inchar='f';
    }
    if (leer== 'g'){
      inchar='g';
    }
    if (leer== 'h'){
      inchar='h';
    }
    if (leer== 'i'){
      inchar='i';
    }
  }
}

```

```

}
if (leer== 'j'){
  inchar='j';
}
if (leer== 'k'){
  inchar='k';
}
if (leer== 'l'){
  inchar='l';
}
if (leer== 'm'){
  inchar='m';
}
if (leer== 'n'){
  inchar='n';
}
if (leer== 'o'){
  inchar='o';
}
if (leer== 'p'){
  inchar='p';
}
if (leer== 'q'){
  inchar='q';
}
if (leer== 'r'){
  inchar='r';
}
if (leer== 's'){
  inchar='s';
}
if (leer== 't'){
  inchar='t';
}
}
if (leer== 'u'){
  inchar='u';
}
if (leer== 'v'){
  inchar='v';
}
if (leer== 'w'){
  inchar='w';
}
if (leer== 'x'){
  inchar='x';
}
if (leer== 'y'){
  inchar='y';
}
if (leer== 'z'){
  inchar='z';
}
if (contadorprecio==1){
  letra25=inchar;
  valor2=letra25;
  convertir2();
  letra25x=letrax2;
  EEPROM.write(25,letra25x);
  contadorprecio --;
}
if (contadorprecio==2){
  letra24=inchar;
  valor2=letra24;
  convertir2();
  letra24x=letrax2;
  EEPROM.write(24,letra24x);
}
contadorprecio --;
}
if (contadorprecio==3){
  letra23=inchar;
  valor2=letra23;
  convertir2();
  letra23x=letrax2;
  EEPROM.write(23,letra23x);
  contadorprecio --;
}
if (contadorprecio==4){
  letra22=inchar;
  valor2=letra22;
  convertir2();
  letra22x=letrax2;
  EEPROM.write(22,letra22x);
  contadorprecio --;
}
if (contadorprecio==5){
  letra21=inchar;
  valor2=letra21;
  convertir2();
  letra21x=letrax2;
  EEPROM.write(21,letra21x);
  contadorprecio --;
}
if (contadorprecio==6){
  letra20=inchar;
  valor2=letra20;
  convertir2();
  letra20x=letrax2;
  EEPROM.write(20,letra20x);
  contadorprecio --;
}

```

```

}
if (contadorprecio==7){
letra19=inchar;
valor2=letra19;
convertir2();
letra19x=letrax2;
EEPROM.write(19,letra19x);
contadorprecio --;
}
if (contadorprecio==8){
letra18=inchar;
valor2=letra18;
convertir2();
letra18x=letrax2;
EEPROM.write(18,letra18x);
contadorprecio --;
}
if (contadorprecio==9){
letra17=inchar;
valor2=letra17;
convertir2();
letra17x=letrax2;
EEPROM.write(17,letra17x);
contadorprecio --;
}
if (contadorprecio==10){
letra16=inchar;
valor2=letra16;
convertir2();
letra16x=letrax2;
EEPROM.write(16,letra16x);
contadorprecio --;
}
if (contadorprecio==11){
letra15=inchar;
valor2=letra15;
convertir2();
letra15x=letrax2;
EEPROM.write(15,letra15x);
contadorprecio --;
}
if (contadorprecio==12){
letra14=inchar;
valor2=letra14;
convertir2();
letra14x=letrax2;
EEPROM.write(14,letra14x);
contadorprecio --;
}
if (contadorprecio==13){
letra13=inchar;
valor2=letra13;
convertir2();
letra13x=letrax2;
EEPROM.write(13,letra13x);
contadorprecio --;
}
if (contadorprecio==14){
letra12=inchar;
valor2=letra12;
convertir2();
letra12x=letrax2;
EEPROM.write(12,letra12x);
contadorprecio --;
}
if (contadorprecio==15){
letra11=inchar;
valor2=letra11;
convertir2();
letra11x=letrax2;
EEPROM.write(11,letra11x);
contadorprecio --;
}
if (contadorprecio==16){
letra10=inchar;
valor2=letra10;
convertir2();
letra10x=letrax2;
EEPROM.write(10,letra10x);
contadorprecio --;
}
if (contadorprecio==17){
letra9=inchar;
valor2=letra9;
convertir2();
letra9x=letrax2;
EEPROM.write(9,letra9x);
contadorprecio --;
}
if (contadorprecio==18){
letra8=inchar;
valor2=letra8;
convertir2();
letra8x=letrax2;
EEPROM.write(8,letra8x);
contadorprecio --;
}
if (contadorprecio==19){
letra7=inchar;

```

```

valor2=letra7;
convertir2();
letra7x=letrax2;
EEPROM.write(7,letra7x);
contadorprecio --;
}
if (contadorprecio==20){
letra6=inchar;
valor2=letra6;
convertir2();
letra6x=letrax2;
EEPROM.write(6,letra6x);
contadorprecio --;
}
if (contadorprecio==21){
letra5=inchar;
valor2=letra5;
convertir2();
letra5x=letrax2;
EEPROM.write(5,letra5x);
contadorprecio --;
}
if (contadorprecio==22){
letra4=inchar;
valor2=letra4;
convertir2();
letra4x=letrax2;
EEPROM.write(4,letra4x);
contadorprecio --;
}
if (contadorprecio==23){
letra3=inchar;
valor2=letra3;
convertir2();
letra3x=letrax2;
EEPROM.write(3,letra3x);
contadorprecio --;
}
if (contadorprecio==24){
letra2=inchar;
valor2=letra2;
convertir2();
letra2x=letrax2;
EEPROM.write(2,letra2x);
contadorprecio --;
}
if (contadorprecio==25){
letra1=inchar;
valor2=letra1;
convertir2();
letra1x=letrax2;
EEPROM.write(1,letra1x);
contadorprecio --;
}
if (contadorprecio==26){
letra0=inchar;
valor2=letra0;
convertir2();
letra0x=letrax2;
EEPROM.write(0,letra0x);
contadorprecio --;
}
}
if (bandera == 4){
if (contadorprecio==0){
bandera=0;
}
if(Serial.available()>0){
leer=Serial.read();
if (leer== '0'){
inchar=0;
}
if (leer== '1'){
inchar=1;
}
if (leer== '2'){
inchar=2;
}
if (leer== '3'){
inchar=3;
}
if (leer== '4'){
inchar=4;
}
if (leer== '5'){
inchar=5;
}
if (leer== '6'){
inchar=6;
}
if (leer== '7'){
inchar=7;
}
if (leer== '8'){
inchar=8;
}
}
}

```



```

}                EEPROM.write(73,digitob2);        contadorprecio --;
if (leer== '9'){   contadorprecio --;                }
    inchar=9;      }
}                if (contadorprecio==3){        digitob5=inchar;
if (contadorprecio==1){   digitob3=inchar;        EEPROM.write(70,digitob5);
digitob1=inchar;        EEPROM.write(72,digitob3);   contadorprecio --;
EEPROM.write(74,digitob1);   contadorprecio --;        }
contadorprecio --;        }
}                if (contadorprecio==4){        }
if (contadorprecio==2){   digitob4=inchar;
digitob2=inchar;        EEPROM.write(71,digitob4);

/*if (tft.height() > 64) {
    for (uint8_t cnt = 0; cnt < 4; cnt++) {
        aspect = (cnt + 0) & 3;
        tft.setRotation(aspect);
        wid = tft.width();
        ht = tft.height();
        msglin = (ht > 160) ? 200 : 112;
        testText();
        dx = wid / 32;
        for (n = 0; n < 32; n++) {
            rgb = n * 8;
            rgb = tft.color565(rgb, rgb, rgb);
            tft.fillRect(n * dx, 48, dx, 63, rgb & colormask[aspect]);
        }
        tft.drawRect(0, 48 + 63, wid, 1, WHITE);
        tft.setTextSize(2);
        tft.setTextColor(colormask[aspect], BLACK);
        tft.setCursor(0, 72);
        tft.print(colormask[aspect]);
        tft.setTextColor(WHITE);
        tft.println(" COLOR GRADES");

```

```

tft.setTextColor(WHITE, BLACK);

printmsg(184, aspectname[aspect]);

delay(1000);

tft.drawPixel(0, 0, YELLOW);

pixel = tft.readPixel(0, 0);

tft.setTextSize((ht > 160) ? 2 : 1); //for messages

#if defined(MCUFRIEND_KBV_H_)

#if 1

extern const uint8_t penguin[];

tft.setAddrWindow(wid - 40 - 40, 20 + 0, wid - 1 - 40, 20 + 39);

tft.pushColors(penguin, 1600, 1);

#elif 1

extern const uint8_t wifi_full[];

tft.setAddrWindow(wid - 40 - 40, 20 + 0, wid - 40 - 40 + 31, 20 + 31);

tft.pushColors(wifi_full, 1024, 1, true);

#elif 1

extern const uint8_t icon_40x40[];

tft.setAddrWindow(wid - 40 - 40, 20 + 0, wid - 1 - 40, 20 + 39);

tft.pushColors(icon_40x40, 1600, 1);

#endif

tft.setAddrWindow(0, 0, wid - 1, ht - 1);

if (aspect & 1) tft.drawRect(wid - 1, 0, 1, ht, WHITE);

else tft.drawRect(0, ht - 1, wid, 1, WHITE);

printmsg(msglin, "VERTICAL SCROLL UP");

uint16_t maxscroll;

if (tft.getRotation() & 1) maxscroll = wid;

else maxscroll = ht;

for (uint16_t i = 1; i <= maxscroll; i++) {

    tft.vertScroll(0, maxscroll, i);

    delay(10);

}

delay(1000);

printmsg(msglin, "VERTICAL SCROLL DN");

```

```

for (uint16_t i = 1; i <= maxscroll; i++) {
    tft.vertScroll(0, maxscroll, 0 - (int16_t)i);
    delay(10);
}

    tft.vertScroll(0, maxscroll, 0);
printmsg(msglin, "SCROLL DISABLED  ");

delay(1000);
if ((aspect & 1) == 0) { //Portrait
    tft.setTextColor(BLUE, BLACK);
    printmsg(msglin, "ONLY THE COLOR BAND");
    for (uint16_t i = 1; i <= 64; i++) {
        tft.vertScroll(48, 64, i);
        delay(20);
    }
    delay(1000);
}
#endif

tft.setTextColor(YELLOW, BLACK);
if (pixel == YELLOW) {
    printmsg(msglin, "SOFTWARE SCROLL  ");
#ifdef 0
    // diagonal scroll of block
    for (int16_t i = 45, dx = 2, dy = 1; i > 0; i -= dx) {
        windowScroll(24, 8, 90, 40, dx, dy, scrollbuf);
    }
#else
    // plain horizontal scroll of block
    n = (wid > 320) ? 320 : wid;
    for (int16_t i = n, dx = 4, dy = 0; i > 0; i -= dx) {
        windowScroll(0, 200, n, 16, dx, dy, scrollbuf);
    }
#endif
}
#endif

```

```

    }
    else if (pixel == CYAN)
        tft.println("readPixel() reads as BGR");
    else if ((pixel & 0xF8F8) == 0xF8F8)
        tft.println("readPixel() should be 24-bit");
    else {
        tft.print("readPixel() reads 0x");
        tft.println(pixel, HEX);
    }
    delay(5000);
}

printmsg(msglin, "INVERT DISPLAY ");
tft.invertDisplay(true);
delay(2000);
tft.invertDisplay(false);
*/
} //llave void loop

typedef struct {
    PGM_P msg;
    uint32_t ms;
} TEST;
TEST result[12];

#define RUNTEST(n, str, test) { result[n].msg = PSTR(str); result[n].ms = test; delay(500); }

void runtests(void)
{
    uint8_t i, len = 24, cnt;
    uint32_t total;
    RUNTEST(0, "FillScreen", testFillScreen());

```

```

RUNTEST(1, "Text          ", testText());
RUNTEST(2, "Lines          ", testLines(CYAN));
RUNTEST(3, "Horiz/Vert Lines    ", testFastLines(RED, BLUE));
RUNTEST(4, "Rectangles (outline) ", testRects(GREEN));
RUNTEST(5, "Rectangles (filled)  ", testFilledRects(YELLOW, MAGENTA));
RUNTEST(6, "Circles (filled)    ", testFilledCircles(10, MAGENTA));
RUNTEST(7, "Circles (outline)   ", testCircles(10, WHITE));
RUNTEST(8, "Triangles (outline)  ", testTriangles());
RUNTEST(9, "Triangles (filled)   ", testFilledTriangles());
RUNTEST(10, "Rounded rects (outline) ", testRoundRects());
RUNTEST(11, "Rounded rects (filled) ", testFilledRoundRects());

tft.fillScreen(BLACK);
tft.setTextColor(GREEN);
tft.setCursor(0, 0);
uint16_t wid = tft.width();
if (wid > 176) {
    tft.setTextSize(2);
#ifdef MCUFRIEND_KBV_H_
    tft.print("MCUFRIEND ");
#endif
#ifdef MCUFRIEND_KBV_H_ != 0
    tft.print(0.01 * MCUFRIEND_KBV_H_, 2);
#else
    tft.print("for");
#endif
    tft.println(" UNO");
#else
    tft.println("Adafruit-Style Tests");
#endif
} else len = wid / 6 - 8;
tft.setTextSize(1);
total = 0;
for (i = 0; i < 12; i++) {

```

```

PGM_P str = result[i].msg;

char c;

if (len > 24) {
    if (i < 10) tft.print(" ");
    tft.print(i);
    tft.print(": ");
}

uint8_t cnt = len;

while ((c = pgm_read_byte(str++)) && cnt--) tft.print(c);

tft.print(" ");

tft.println(result[i].ms);

total += result[i].ms;
}

tft.setTextSize(2);

tft.print("Total:");

tft.print(0.000001 * total);

tft.println("sec");

g_identifier = tft.readID();

tft.print("ID: 0x");

tft.println(tft.readID(), HEX);

// tft.print("Reg(00):0x");

// tft.println(tft.readReg(0x00), HEX);

tft.print("F_CPU:");

tft.print(0.000001 * F_CPU);

#ifdef __OPTIMIZE_SIZE__

    tft.println("MHz -Os");

#else

    tft.println("MHz");

#endif

delay(10000);
}

```

```
// Standard Adafruit tests. will adjust to screen size
```

```
unsigned long testFillScreen() {
    unsigned long start = micros();
    tft.fillScreen(BLACK);
    tft.fillScreen(RED);
    tft.fillScreen(GREEN);
    tft.fillScreen(BLUE);
    tft.fillScreen(BLACK);
    return micros() - start;
}

unsigned long testText() {
    unsigned long start;
    tft.fillScreen(BLACK);
    start = micros();
    tft.setCursor(0, 0);
    tft.setTextColor(WHITE); tft.setTextSize(1);
    tft.println("Hello World!");
    tft.setTextColor(YELLOW); tft.setTextSize(2);
    tft.println(123.45);
    tft.setTextColor(RED); tft.setTextSize(5);
    tft.println(0xDEADBEEF, HEX);
    tft.println();
    tft.setTextColor(GREEN);

    x1 = y1 = 0;
    y2 = h - 1;
    start = micros();
    for (x2 = 0; x2 < w; x2 += 6) tft.drawLine(x1, y1, x2, y2, color);
    x2 = w - 1;
    for (y2 = 0; y2 < h; y2 += 6) tft.drawLine(x1, y1, x2, y2, color);
    t = micros() - start; // fillScreen doesn't count against timing

    tft.setTextSize(5);
    tft.println("Groop");
    tft.setTextSize(2);
    tft.println("I implore thee,");
    tft.setTextSize(1);
    tft.println("my foonting turlingdromes.");
    tft.println("And hooptiously drangle me");
    tft.println("with crinkly bindlewurdles,");
    tft.println("Or I will rend thee");
    tft.println("in the gobberwarts");
    tft.println("with my blurglecruncheon,");
    tft.println("see if I don't!");
    return micros() - start;
}

unsigned long testLines(uint16_t color) {
    unsigned long start, t;
    int x1, y1, x2, y2,
        w = tft.width(),
        h = tft.height();

    tft.fillScreen(BLACK);
```

```
tft.fillScreen(BLACK);

x1 = w - 1;
y1 = 0;
y2 = h - 1;
start = micros();
for (x2 = 0; x2 < w; x2 += 6) tft.drawLine(x1, y1, x2, y2, color);
x2 = 0;
for (y2 = 0; y2 < h; y2 += 6) tft.drawLine(x1, y1, x2, y2, color);
t += micros() - start;
```

```
tft.fillScreen(BLACK);

x1 = 0;
y1 = h - 1;
y2 = 0;
start = micros();
for (x2 = 0; x2 < w; x2 += 6) tft.drawLine(x1, y1, x2, y2, color);
x2 = w - 1;
for (y2 = 0; y2 < h; y2 += 6) tft.drawLine(x1, y1, x2, y2, color);
t += micros() - start;
```

```
tft.fillScreen(BLACK);

x1 = w - 1;
y1 = h - 1;
y2 = 0;
start = micros();
for (x2 = 0; x2 < w; x2 += 6) tft.drawLine(x1, y1, x2, y2, color);
x2 = 0;
for (y2 = 0; y2 < h; y2 += 6) tft.drawLine(x1, y1, x2, y2, color);
```



```

    return micros() - start;
}

unsigned long testFastLines(uint16_t color1, uint16_t color2) {
    unsigned long start;

    int    x, y, w = tft.width(), h = tft.height();

    tft.fillScreen(BLACK);

    start = micros();
    for (y = 0; y < h; y += 5) tft.drawFastHLine(0, y, w, color1);
    for (x = 0; x < w; x += 5) tft.drawFastVLine(x, 0, h, color2);

    return micros() - start;
}

unsigned long testRects(uint16_t color) {
    unsigned long start;

    int    n, i, i2,
           cx = tft.width() / 2,
           cy = tft.height() / 2;

    tft.fillScreen(BLACK);

    n    = min(tft.width(), tft.height());
    start = micros();

    for (i = 2; i < n; i += 6) {
        i2 = i / 2;
        tft.drawRect(cx - i2, cy - i2, i, i, color);
    }

    return micros() - start;
}

unsigned long testFilledRects(uint16_t color1, uint16_t color2) {

```

```

unsigned long start, t = 0;

int    n, i, i2,
        cx = tft.width() / 2 - 1,
        cy = tft.height() / 2 - 1;

tft.fillScreen(BLACK);

n = min(tft.width(), tft.height());
for (i = n; i > 0; i -= 6) {
    i2  = i / 2;
    start = micros();
    tft.fillRect(cx - i2, cy - i2, i, i, color1);
    t    += micros() - start;
    // Outlines are not included in timing results
    tft.drawRect(cx - i2, cy - i2, i, i, color2);
}

return t;
}

unsigned long testFilledCircles(uint8_t radius, uint16_t color) {
    unsigned long start;
    int x, y, w = tft.width(), h = tft.height(), r2 = radius * 2;

    tft.fillScreen(BLACK);
    start = micros();
    for (x = radius; x < w; x += r2) {
        for (y = radius; y < h; y += r2) {
            tft.fillCircle(x, y, radius, color);
        }
    }

    return micros() - start;
}

```

```

unsigned long testCircles(uint8_t radius, uint16_t color) {
    unsigned long start;
    int    x, y, r2 = radius * 2,
           w = tft.width() + radius,
           h = tft.height() + radius;

    // Screen is not cleared for this one -- this is
    // intentional and does not affect the reported time.
    start = micros();
    for (x = 0; x < w; x += r2) {
        for (y = 0; y < h; y += r2) {
            tft.drawCircle(x, y, radius, color);
        }
    }

    return micros() - start;
}

unsigned long testTriangles() {
    unsigned long start;
    int    n, i, cx = tft.width() / 2 - 1,
           cy = tft.height() / 2 - 1;

    tft.fillScreen(BLACK);
    n = min(cx, cy);
    start = micros();
    for (i = 0; i < n; i += 5) {
        tft.drawTriangle(
            cx , cy - i, // peak
            cx - i, cy + i, // bottom left
            cx + i, cy + i, // bottom right
            tft.color565(0, 0, i));
    }
}

```

```

    }

    return micros() - start;
}

unsigned long testFilledTriangles() {
    unsigned long start, t = 0;
    int    i, cx = tft.width() / 2 - 1,
           cy = tft.height() / 2 - 1;

    tft.fillScreen(BLACK);
    start = micros();
    for (i = min(cx, cy); i > 10; i -= 5) {
        start = micros();
        tft.fillTriangle(cx, cy - i, cx - i, cy + i, cx + i, cy + i,
                        tft.color565(0, i, i));
        t += micros() - start;
        tft.drawTriangle(cx, cy - i, cx - i, cy + i, cx + i, cy + i,
                        tft.color565(i, i, 0));
    }

    return t;
}

unsigned long testRoundRects() {
    unsigned long start;
    int    w, i, i2, red, step,
           cx = tft.width() / 2 - 1,
           cy = tft.height() / 2 - 1;

    tft.fillScreen(BLACK);
    w = min(tft.width(), tft.height());
    start = micros();

```

```
red = 0;
step = (256 * 6) / w;
for (i = 0; i < w; i += 6) {
    i2 = i / 2;
    red += step;
    tft.drawRoundRect(cx - i2, cy - i2, i, i, i / 8, tft.color565(red, 0, 0));
}
```

```

return micros() - start;
}

unsigned long testFilledRoundRects() {
    unsigned long start;

    int    i, i2, green, step,
           cx = tft.width() / 2 - 1,
           cy = tft.height() / 2 - 1;

    tft.fillScreen(BLACK);

    start = micros();

    green = 256;

    step = (256 * 6) / min(tft.width(), tft.height());

    for (i = min(tft.width(), tft.height()); i > 20; i -= 6) {
        i2 = i / 2;

        green -= step;

        tft.fillRoundRect(cx - i2, cy - i2, i, i, i / 8, tft.color565(0, green, 0));
    }

    return micros() - start;
}

void convertir (Evans,
2011){
    if (letrax ==97){
        valor='a';
    }
    if (letrax ==98){
        valor='b';
    }
    if (letrax ==99){
        valor='c';
    }
    if (letrax ==100){
        valor='d';
    }
    if (letrax ==101){
        valor='e';
    }
    if (letrax ==102){
        valor='f';
    }
    if (letrax ==103){
        valor='g';
    }
    if (letrax ==104){
        valor='h';
    }
    if (letrax ==105){
        valor='i';
    }
    if (letrax ==106){
        valor='j';
    }
    if (letrax ==107){
        valor='k';
    }
    if (letrax ==108){

```

```

    valor='l';
}
if (letrax ==109){
    valor='m';
}
if (letrax ==110){
    valor='n';
}
if (letrax ==111){
    valor='o';
}
if (letrax ==112){
    valor='p';
}
if (letrax ==113){
    valor='q';
}
if (letrax ==114){
    valor='r';
}
if (letrax ==115){
    valor='s';
}
if (letrax ==116){
    valor='t';
}
if (letrax ==117){
    valor='u';
}
if (letrax ==118){
    valor='v';
}
if (letrax ==119){
    valor='w';
}
if (letrax ==120){
    valor='x';
}
if (letrax ==121){
    valor='y';
}
if (letrax ==122){
    valor='z';
}
if (letrax ==32){
    valor=' ';
}
void convertir2(){
    if (valor2=='a'){
        letrax2=97;
    }
    if (valor2=='b'){
        letrax2=98;
    }
    if (valor2=='c'){
        letrax2=99;
    }
    if (valor2=='d'){
        letrax2=100;
    }
    if (valor2=='e'){
        letrax2=101;
    }
    if (valor2=='f'){
        letrax2=102;
    }
    if (valor2=='g'){
        letrax2=103;
    }
    if (valor2=='h'){
        letrax2=104;
    }
    if (valor2=='i'){
        letrax2=105;
    }
    if (valor2=='j'){
        letrax2=106;
    }
    if (valor2=='k'){
        letrax2=107;
    }
    if (valor2=='l'){
        letrax2=108;
    }
    if (valor2=='m'){
        letrax2=109;
    }
    if (valor2=='n'){
        letrax2=110;
    }
    if (valor2=='o'){
        letrax2=111;
    }
    if (valor2=='p'){
        letrax2=112;
    }
    if (valor2=='q'){
        letrax2=113;
    }
}

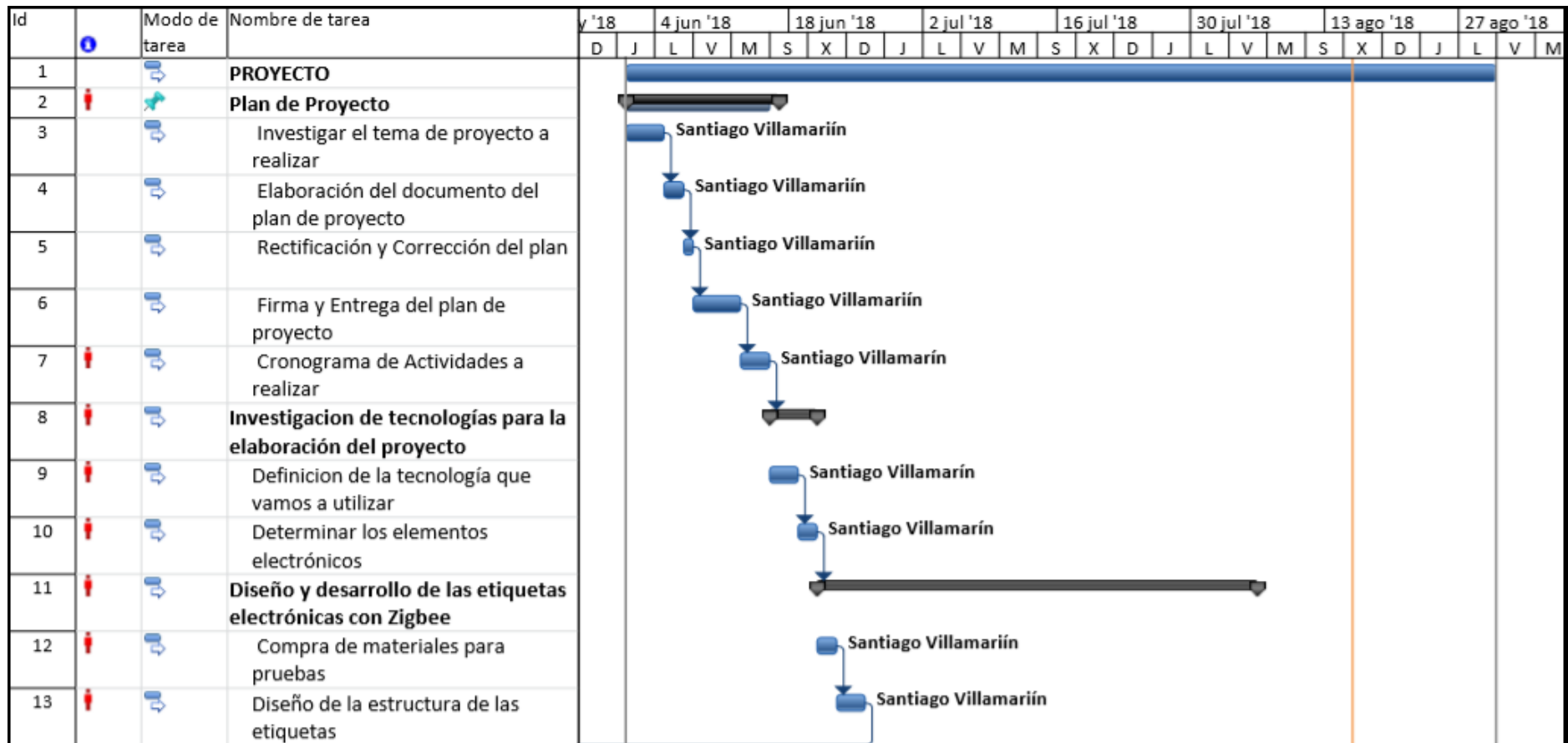
```

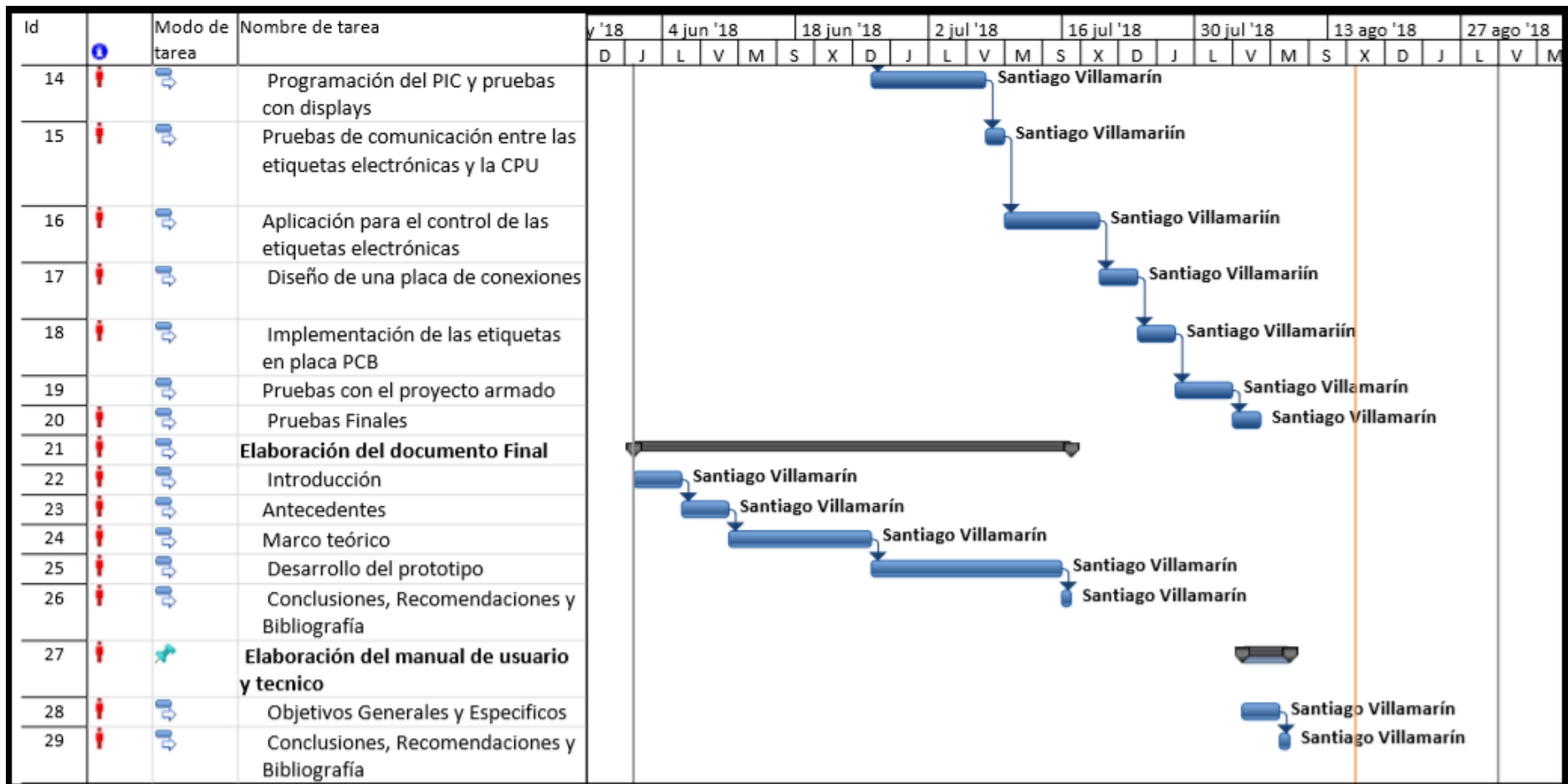
```

}
if (valor2=='r'){
letrax2=114;
}
if (valor2=='s'){
letrax2=115;
}
if (valor2=='t'){
letrax2=116;
}
if (valor2=='u'){
letrax2=117;
}
if (valor2=='v'){
letrax2=118;
}
if (valor2=='w'){
letrax2=119;
}
if (valor2=='x'){
letrax2=120;
}
if (valor2=='y'){
letrax2=121;
}
if (valor2=='z'){
letrax2=122;
}
if (valor2==' '){
letrax2=32;
}
/

```

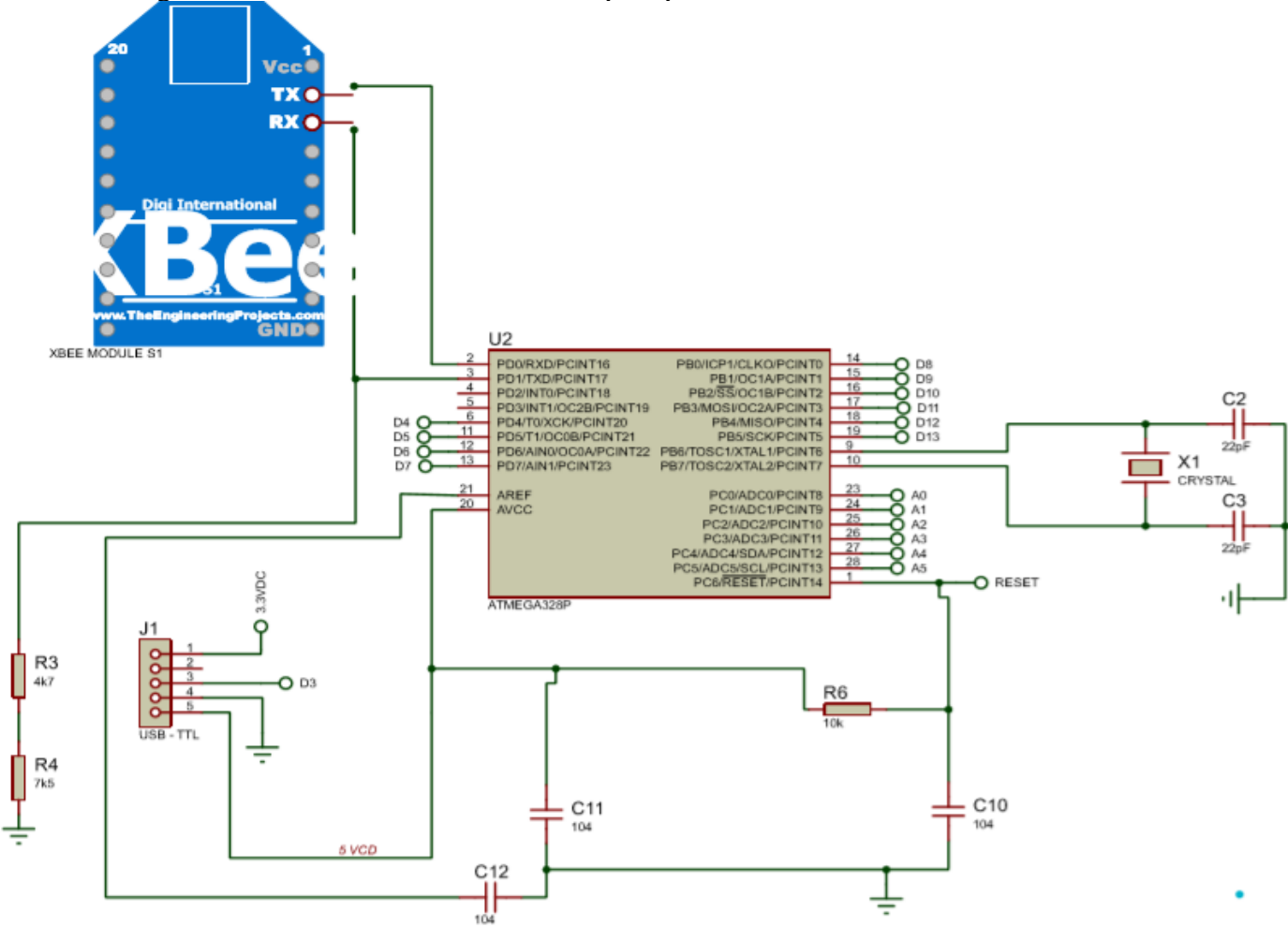

Anexo 5: Cronograma de actividades.





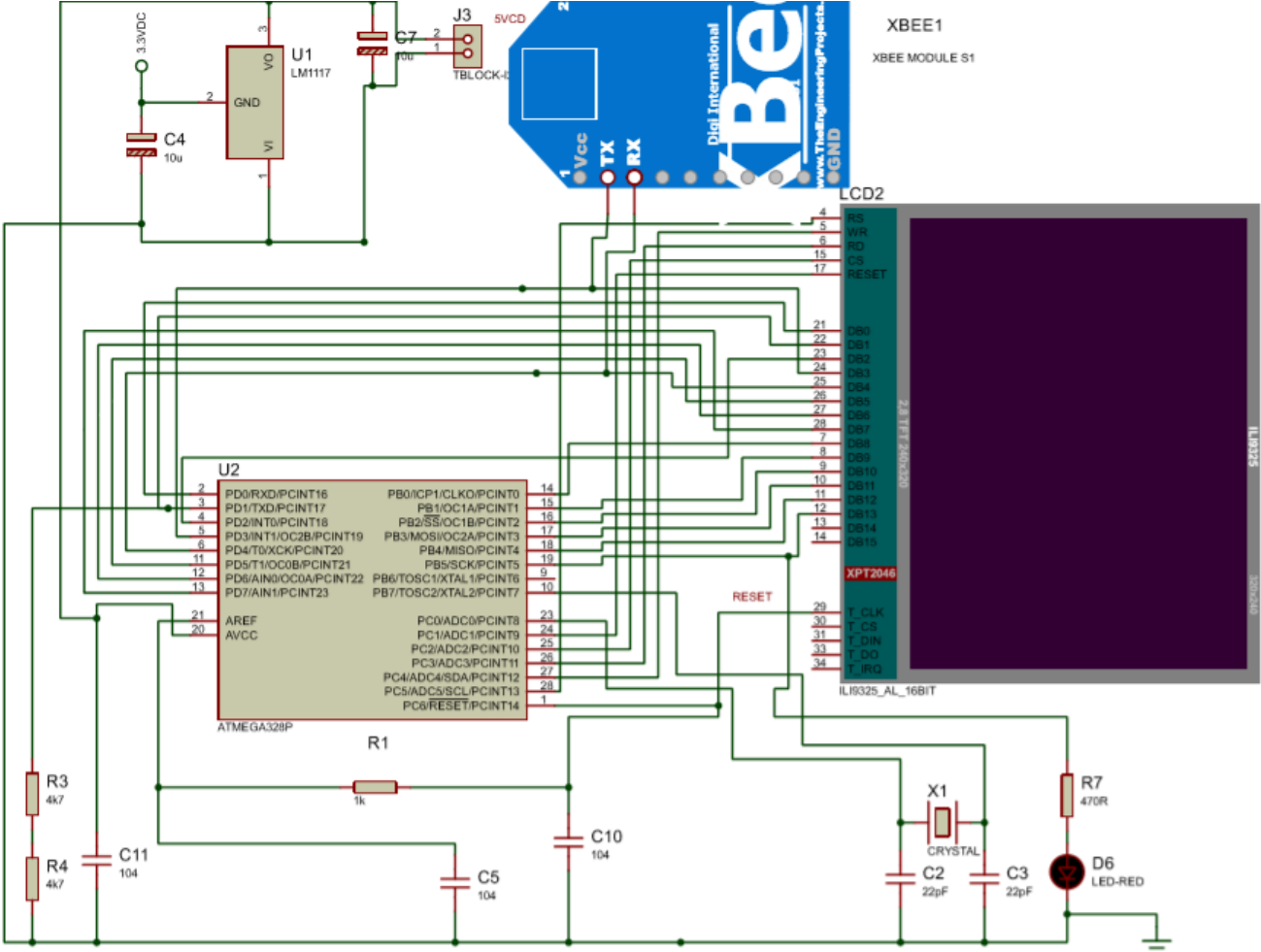
Fuente:(Elaborado por el autor)

Anexo 6: Diagrama electrónico de la controladora principal



Fuente:(Elaborado por el autor)

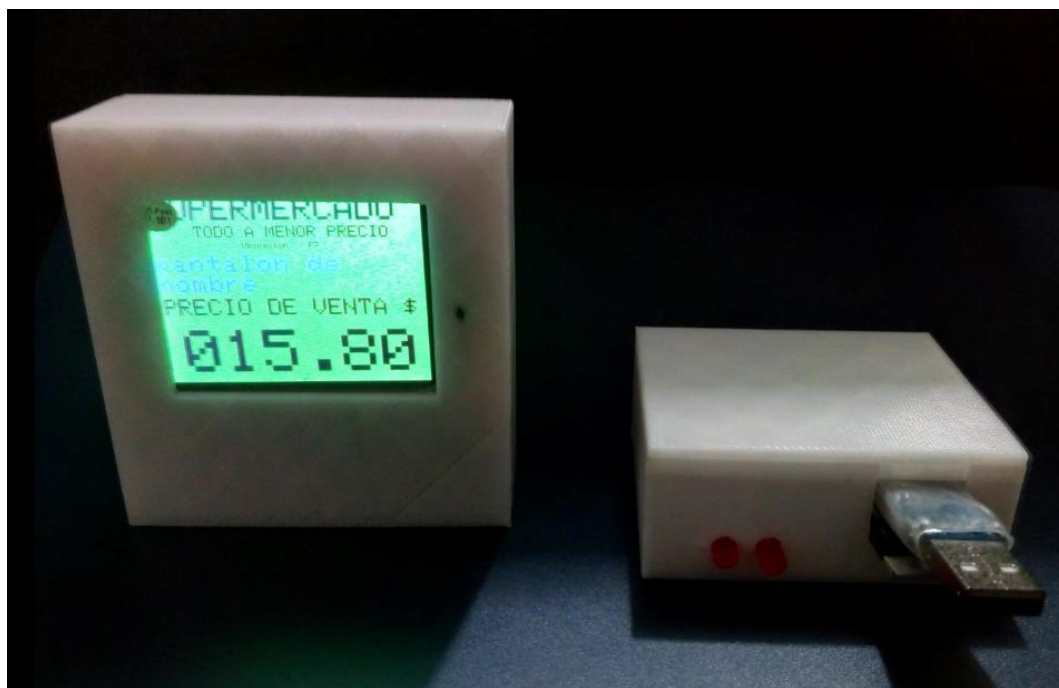
Anexo 7: Diagrama etiqueta Electronica







Fuente:(Elaborado por el autor)

Anexo 8: MANUAL DE USUARIO

Etiquetas electrónicas inalámbricas con tecnología Zigbee para Supermercados.



Quito, 13 de agosto de 2018

	<p>No manipular la tarjeta electrónica.</p>
	<p>No desconectar la controladora principal cuando este transmitiendo datos.</p>
	<p>No golpear el dispositivo o placa electrónica.</p>
	<p>No exponer el dispositivo electrónico al agua.</p>

Introducción.

Las etiquetas electrónicas tienen como objetivo presentar los precios y descripción de los productos de un Supermercado de una manera más llamativa y práctica.

El sistema consta de 2 módulos. La controladora principal y las tarjetas electrónicas.

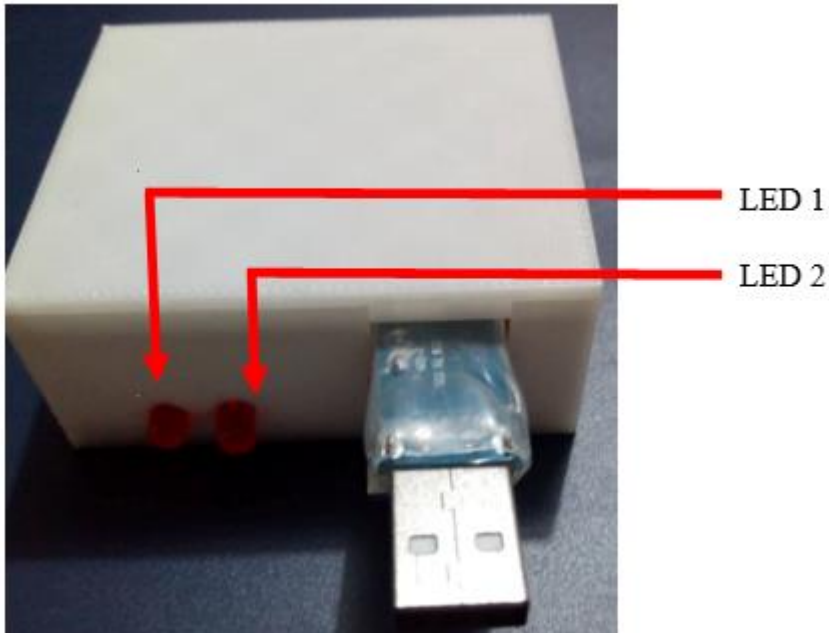
La controladora principal que es el coordinador Zigbee se encarga de comunicarse con las etiquetas electrónicas para modificar su precio y descripción.

La etiqueta electrónica es la que presenta por medio de un display información del producto que se encuentra en la percha.

Las etiquetas una vez que se estableció su precio y descripción pueden trabajar independientemente de la controladora principal.

1) Requerimientos Técnicos del Sistema e Instalación.

- **CONTROLADORA PRINCIPAL**



Descripción de los Leds

Nombre	Estatus	Descripción
Energía suministrada de puerto USB de PC	Encendido (titilando)	Controladora principal encendida.
	Apagado	Controladora principal apagada.
Led 1	titilando	Habilitada controladora para comunicación.
Led 2	enciende	Está enviando datos hacia la etiqueta electrónica.

Instalación:

La controladora principal debe instalarse en una computadora con sistema operativo Windows preferible 10.

Antes de conectar la controladora principal se realiza unos cambios en nuestra PC:

Cargar el driver recomendado para el convertidor USB-TTL. Se lo puede bajar del siguiente enlace: http://www.prolific.com.tw/US/ShowProduct.aspx?p_id=225&pcid=41

Conectar la controladora principal y en Control panel/administrador de dispositivos cambiar en puerto serial a COM1.

Copiar la carpeta **Aplicación etiquetas electrónicas** ubicado en el CD de instalación.

Ejecutar el programa ejecutable Proyecto 1.



1. Cargar previamente el driver TTL y modificar a COM1.
2. Conectar la controladora principal.
3. Abrir la aplicación Proyecto1.
4. Se puede modificar las etiquetas electrónicas.

- **ETIQUETAS ELECTRÓNICAS.**

Para instalar las etiquetas electrónicas, se debe colocarlas en el lugar que corresponden con el producto y poner su switch en ON.

2 Pruebas de funcionamiento.

Para probar el funcionamiento del sistema, se tiene la aplicación abierta y las etiquetas electrónicas en modo ON.

Aplicación.



Una vez abierta la aplicación se puede modificar la información de las etiquetas electrónicas.

Solo se debe dar clic en Modificar precio, nombre o modificar precio oferta de la etiqueta que se desee cambiar.

Se ingresa la información que se necesita modificar, el led 2 se enciende y automáticamente se cambia la información de la etiqueta.

Nota. Los valores a ingresar para modificar la especificación del producto tienen un número determinado de caracteres, rellenar con espacios para que transmita más rápido.

Etiquetas electrónicas.

Modificar las etiquetas electrónicas.

Una vez abierto la aplicación en la PC, verificar que las etiquetas electrónicas se encuentran encendidas

Valores que se pueden modificar:

Se tiene valores que solo se puede modificar via programación y los valores que se encuentran habilitados para el usuario. En la figura 7.4 y figura 7.5 se observan los valores que se puede modificar por el usuario.

En la descripción del producto podemos poner cualquier palabra dependiendo del producto a etiquetar. En el precio también se puede modificar los valores con dos grados decimales y hasta la centena en la parte entera. Con estos rangos se tendría la totalidad de la variación

de los precios de los productos que se exhiben en un supermercado



Figura 7.1: Pantalla etiqueta electrónica
Fuente: (Elaborado por el autor)

Al escoger la opción de oferta se despliega la información y el valor del producto en oferta. El valor del producto de oferta se lo puede modificar con ayuda de la aplicación.



Figura 7.2: Pantalla etiqueta electrónica.
Fuente:(Elaborado por el autor)

Guía rápida de fallas y soluciones.

Antes de acudir a personal autorizado se recomienda consultar la guía rápida de fallas.

Problema	Solución
No se enciende la etiqueta.	Revisar que la batería no se encuentre descargada o dañada, ponerle a cargar y esperar unos minutos.
La etiqueta se inhibe.	Dar un reset a la etiqueta electrónica.
No se puede enviar datos desde la aplicación.	Revisar que el LED de la controladora este encendido y parpadeando.
Cayo agua a la etiqueta.	Apagarla y llevarla a servicio técnico.
Aplicación da error de COM1	Volver a instalar el driver TTL-USB y configurarle.
No se refleja la información en el display de la etiqueta.	Asegurarse de llenar todos los espacios en la aplicación, ya que si no se llenan la información no se Visualiza.

Contacto de soporte técnico.

Correo electrónico

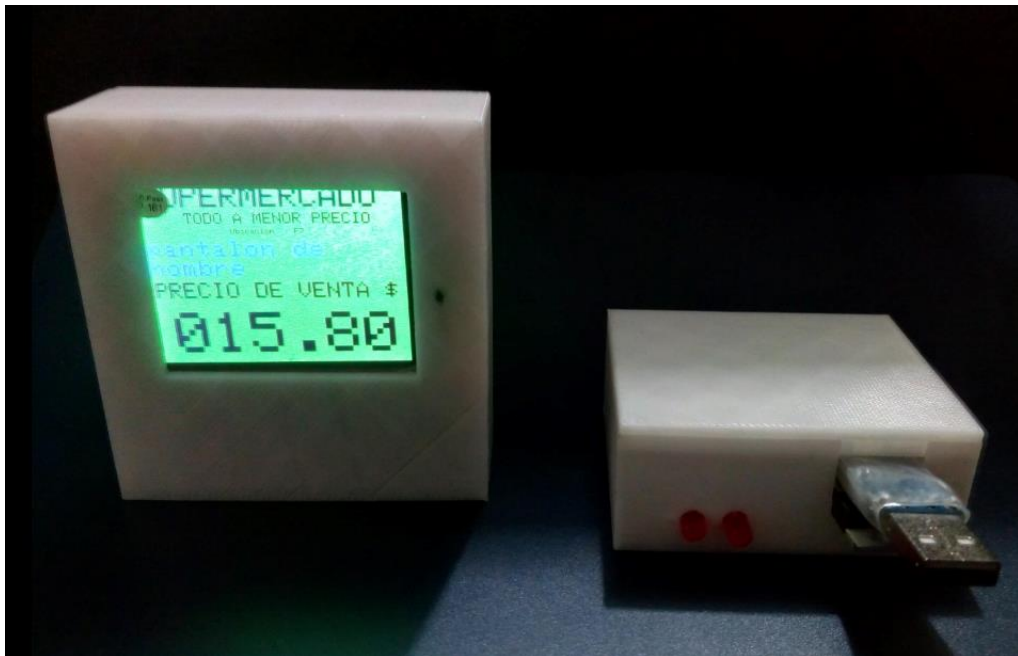
Santiago Villamarín stigodav1@gmail.com

Teléfono





Santiago Villamarín +593 987 271 606

Anexo 9: MANUAL TÉCNICO

Etiquetas electrónicas inalámbricas con tecnología Zigbee para Supermercados.



Quito, 13 de agosto de 2018.

	<p>No manipular la tarjeta electrónica.</p>
	<p>No utilizar baterías, en mal estado en las etiquetas electrónicas. No conectar o desconectar la controladora cuando está transmitiendo.</p>
	<p>No golpear el dispositivo o placa electrónica.</p>
	<p>No exponer el dispositivo electrónico al agua.</p>

Introducción

Las etiquetas electrónicas tienen como objetivo presentar los precios y descripción de los productos de un Supermercado de una manera más llamativa y práctica.

Características

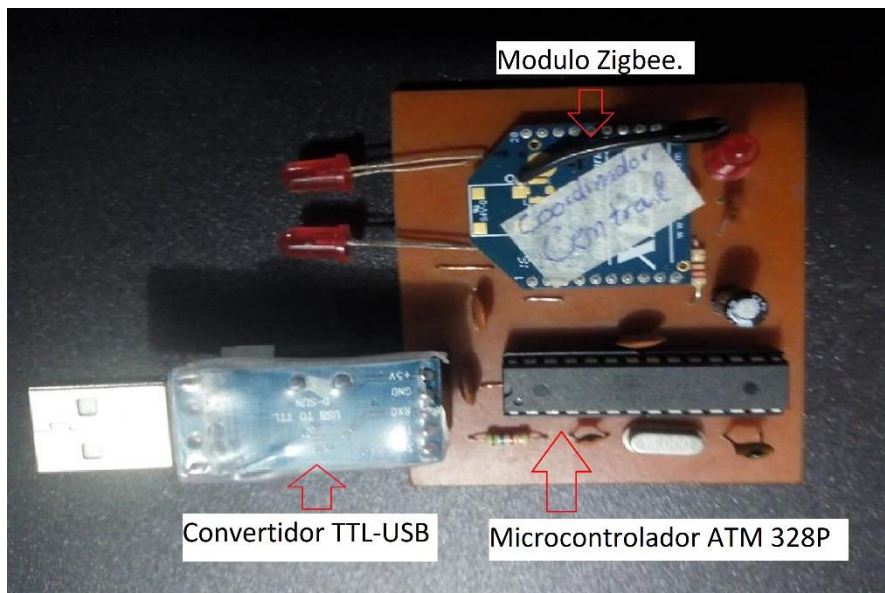
- Aplicación compatible con Windows 7,8,10 solo hay que tener el driver del convertidor TTL a USB.
- Alimentación 5V la tarjeta controladora principal y una batería de 3.7 v 400mA las etiquetas electrónicas
- Cuenta con una pantalla GLDC que muestra información de un producto.

Especificaciones:

Parámetros	Mínimo	Máximo	Batería
Voltaje de entrada Controladora principal	5v	5.5v	
Voltaje de entrada etiqueta electrónica	3,7v	3,7v	Usa una batería de 3.7v a 400ma
Pantalla GLCD			Para Arduino.

Descripción de los circuitos electrónicos

Controlador principal.

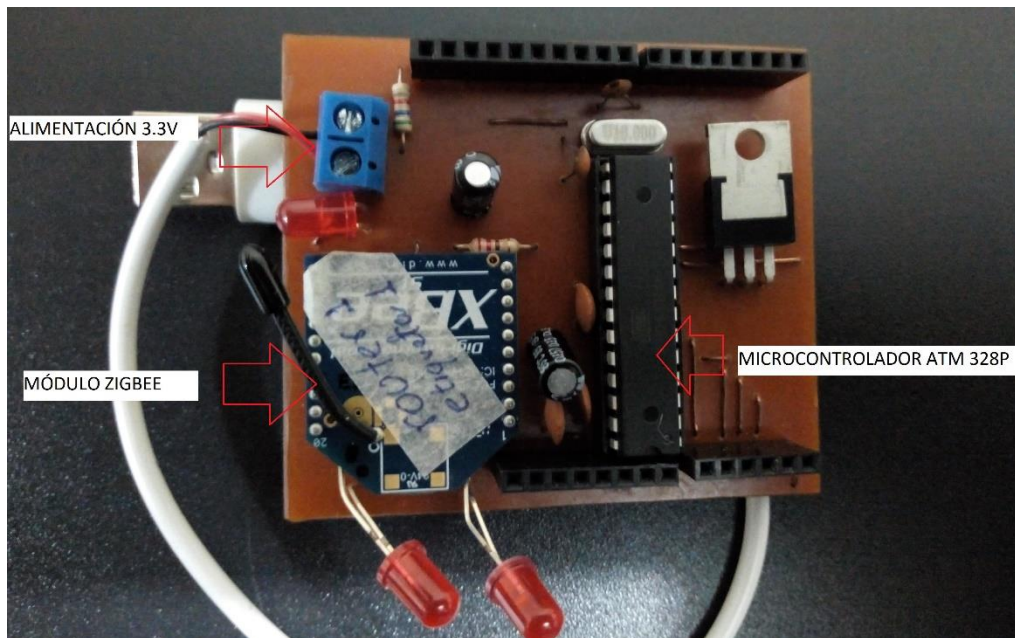


En la controladora principal se puede reemplazar el microcontrolador y el convertidor TTL-USB si estos llegan a dañarse, estos módulos están colocados sobre sócalos.

El microcontrolador para ser reemplazado necesita que se cargue la programación, el reemplazó se puede pedir y adquirir al fabricante.

El convertidor no necesita programación, solo se lo saca y se lo reemplaza.

Etiquetas electrónicas.



En la tarjeta de las etiquetas electrónicas, podemos reemplazar el display GLCD , el microcontrolador y la batería.

El equipo usa una batería de 4,2v 600ma, su tiempo de vida estimado es de 1 año, luego de lo cual debe ser reemplazada.

Requerimientos Técnicos del Sistema, Instalación y Configuración

Conexión.

Disponer de una PC con sistema operativo Windows 10 y con puerto USB en buen estado.

Descripción de los Led

Nombre	Estatus	Descripción
Energía suministrada de puerto USB de PC	Encendido (titilando)	Controladora principal encendida.
	Apagado	Controladora principal apagada.
Led 1	titilando	Habilitada controladora para comunicación.
Led 2	enciende	Está enviando datos hacia la etiqueta electrónica.

Instalación:

La controladora principal debe instalarse en una computadora con sistema operativo Windows preferible 10.

Antes de conectar la controladora principal debemos realizar unos cambios en nuestra PC:

- Cargar el driver recomendado para el convertidor USB-TTL. Se lo puede bajar del siguiente enlace: http://www.prolific.com.tw/US/ShowProduct.aspx?p_id=225&pcid=41
- Conectar la controladora principal y en Control panel/administrador de dispositivos cambiar en puerto serial a COM1.
- Copiar la carpeta **Aplicación etiquetas electrónicas** ubicado en el CD de instalación.
- Ejecutar el programa ejecutable Proyecto 1.



1. Cargar previamente el driver TTL y modificar a COM1.

2. Conectar la controladora principal.
3. Abrir la aplicación Proyecto1.
4. Se puede modificar las etiquetas electrónicas.

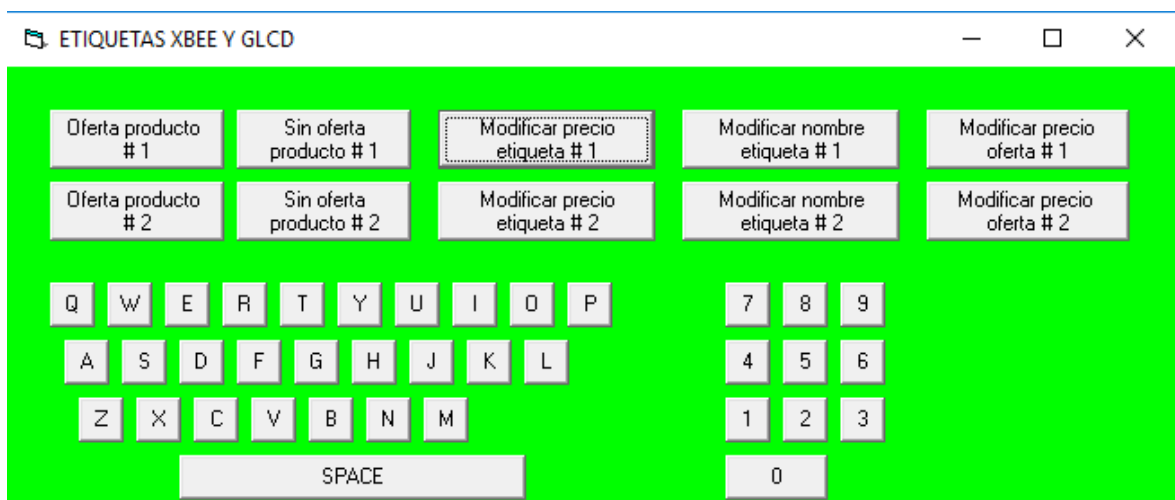
Modificar las etiquetas electrónicas.

Una vez abierto la aplicación en la PC proyecto uno, verificar que las etiquetas electrónicas se encuentran encendidas



Pruebas de funcionamiento.

Uso de la aplicación.



Una vez abierta la aplicación podemos modificar la información de las etiquetas electrónicas.

Solo se debe dar clic en Modificar precio, nombre o modificar precio oferta de la etiqueta que deseemos cambiar.

Se ingresa la información que se necesita modificar, el led 2 se enciende y automáticamente se cambia la información de la etiqueta.

Nota. Los valores a ingresar para modificar el nombre tienen un número determinado de caracteres, rellenar con espacios para que transmita más rápido.

Herramientas para el técnico:

El nombre del Supermercado y la ubicación del equipo solo puede cambiarse por personal autorizado en el programa de Arduino. Hay que encontrar a ubicación, cambiarle y volverle a cambiar al dispositivo. Esto es una medida de seguridad para que el nombre del local no sea fácil de cambiar ante robos.

```
void loop(void) {
  if (bandera==0){

    tft.fillScreen(GREEN);
    tft.setCursor(5, 5);
    tft.setTextColor(BLACK);
    tft.setTextSize(4);
    tft.println("SUPERMERCADO");

    tft.setCursor(55, 40);
    tft.setTextSize(2);
    tft.setTextColor(BLACK);
    tft.println("TODO A MENOR PRECIO");

    tft.setCursor(110, 60);
```

Solo se cambia el nombre entre comillas.

Para agregar más etiquetas a la aplicación necesariamente hay que aumentar

variables a la aplicación en Visual Basic y poner el programa Arduino en la nueva etiqueta.

Guía de fallas y soluciones.

Antes de acudir a personal autorizado se recomienda consultar la guía rápida de fallas.

Problema.	Solución.
No se enciende la etiqueta.	Podemos medir el voltaje y la corriente suministrada por la batería para descartarla.
La etiqueta se inhibe.	Dar un reset a la etiqueta electrónica.
No se puede enviar datos desde la aplicación.	Probar la transmisión entre módulos Zigbee.
Cayo agua a la etiqueta.	Dejarla secar antes de probar y evaluar los daños.
Aplicación da error de COM1	Revisar configuración de USB direccionado con puerto COM1
No se refleja la información en el display de la etiqueta.	Asegurarse de llenar todos los espacios en la aplicación, ya que si no se llenan la información no se Visualiza.

Contacto de soporte técnico.

Correo electrónico

Santiago Villamarín stigodav1@gmail.com

Teléfono

Santiago Villamarín +593 987 271 606

ANTIPLAGIO

7.9% PlagScan Resultados del Análisis de los plagios del 18/08/2018 3:57 **ETIQUETAS ELECTRONICAS FINAL antip2.doc** Fecha: 18/08/2018 3:53

Vista: **Todas las fuentes** 68 121 resultados

- Todas las fuentes 68
- Top tres 3
- Fuentes de internet 68

- [0] <https://github.com/prenticedavid/MCU> 3.6% 54 resultados Marcar resultados en
- [1] <https://forum.kerbalspaceprogram.com> 3.3% 48 resultados Marcar resultados en
1 documento con coincidencias exactas
- [3] https://github.com/prenticedavid/TFT_ 3.0% 41 resultados Marcar resultados en
- [4] [forum.arduino.cc/index.php?topic=44:](http://forum.arduino.cc/index.php?topic=44) 2.7% 35 resultados Marcar resultados en
- [5] <https://github.com/paramaggarwal/tftk> 2.2% 29 resultados Marcar resultados en
- [6] <https://forum.arduino.cc/index.php?to> 1.9% 23 resultados Marcar resultados en

leyenda marcado del texto

- Aa** concordancia exacta
- Aa** cambios del texto posibles
- Aa** marcado como cita



UNIVERSIDAD TECNOLÓGICA ISRAEL

TRABAJO DE TITULACIÓN EN OPCIÓN AL GRADO DE:

“INGENIERO EN ELECTRÓNICA DIGITAL Y TELECOMUNICACIONES”

1 / 78

DECLARACIÓN Y AUTORIZACIÓN.

Yo, Santiago David Villamarín Aguirre, CI 1716030539 autor del trabajo de graduación:

Diseño y desarrollo de etiquetas electrónicas para productos en supermercados, mediante tecnología Zigbee, previo a la obtención del título de Ingeniero en Electrónica Digital y Telecomunicaciones en la UNIVERSIDAD TECNOLÓGICA ISRAEL.

1. Declaro tener pleno conocimiento de la obligación que tienen las instituciones de educación superior, de conformidad con el artículo 144 de la Ley Orgánica de Educación Superior, de difundir el respectivo trabajo de graduación para que sea integrado al Sistema Nacional de Información de la Educación Superior del Ecuador para su difusión pública respetando los derechos de autor.

2. Autorizó a la SENESCYT a tener una copia del referido trabajo de graduación, con el propósito de generar un repositorio que democratice la información, respetando las políticas de propiedad intelectual vigente

Quito, Septiembre del 2018.

Atentamente.



Santiago David Villamarín Aguirre.
C.I. 1716030539