

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

DEPARTAMENTO DE ELECTRÓNICA

VALPARAÍSO – CHILE



CONTROL DE OCUPACION DE CABAÑAS MEDIANTE UNA WPAN

RAMIRO DANIEL QUIROZ SALINAS

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO DE
EJECUCIÓN EN ELECTRÓNICA

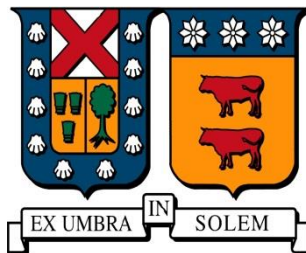
PROFESOR GUÍA: AGUSTÍN GONZÁLEZ VALENZUELA

MAYO - 2015

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

DEPARTAMENTO DE ELECTRÓNICA

VALPARAÍSO – CHILE



CONTROL DE OCUPACION DE CABAÑAS MEDIANTE UNA WPAN

RAMIRO DANIEL QUIROZ SALINAS

**MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO
DE EJECUCIÓN EN ELECTRÓNICA**

PROFESOR GUÍA:

AGUSTÍN GONZÁLEZ VALENZUELA

PROFESOR CORREFERENTE:

DANIEL RODRÍGUEZ

MAYO – 2015

Material de referencia, su uso no involucra responsabilidad del autor o de la Institución

Control de ocupación de cabañas mediante una WPAN

Memoria para optar al título de Ingeniero de ejecución en Electrónica, mención en telecomunicaciones y sistemas digitales

Ramiro Daniel Quiroz Salinas

Profesor Guía: Agustín González

Profesor Correferente: Daniel Rodríguez

Mayo 2015

Resumen

El presente trabajo describe las tecnologías necesarias para montar una WPAN (Wireless Personal Area Network) de baja velocidad de transmisión, corto alcance, y mínima potencia de transmisión, orientada a la telemetría de señales obtenidas por diversos nodos de comunicaciones (End-Device), distribuidos en un conjunto de cabañas, de forma de poder recopilar estas mediciones en un nodo central, que recibirá dicha información desde los End-Devices y la entregará a un servicio de medición de uso de las cabañas ubicado en internet.

La tecnología de transmisión inalámbrica utilizada en este trabajo es el protocolo Miwi, propietario de Microchip, el cual está basado en el protocolo IEEE 802.15.4, y opera en el espectro de frecuencias de 2,4 GHz.

La lógica de control del sistema está basada en el chip 18F27J53 de la familia PIC18 de microcontroladores de 8 bit de microchip, el cual es programado usando el lenguaje C, bajo el ambiente de desarrollo MPLAB X, usando el stack del protocolo antes indicado.

Como herramienta de programación y debugging se ha elegido ICD3 del mismo proveedor, de forma de facilitar el proceso de depuración de las aplicaciones y de minimizar el TTM (Time to market), asunto de vital importancia en cualquier desarrollo tecnológico hoy en día.

En resumen, reuniendo todos estos elementos y tecnologías, se logra un diseño simple, económico, de bajo consumo y gran flexibilidad a la hora de implementar la solución para la problemática de controlar de forma automática la ocupación de las cabañas, dejando al dueño mayor tiempo libre para otras labores, minimizando las pérdidas ocasionadas por el elusión de pagos recibidos desde el administrador local.

La aplicación Web que recibe y procesa esta información está fuera del alcance del presente trabajo, y solo será presentada una versión de prueba simple que demuestra que la información enviada desde los sensores es recibida y almacenada.

Palabras clave: telemetría, wpan, red de área personal.

Control de ocupación de cabañas mediante una WPAN

Undergraduate Thesis for the degree of Ingeniero de ejecución en electrónica, mención en telecomunicaciones y sistemas digitales

Ramiro Daniel Quiroz Salinas

Guide Profesor: Agustín González

Coreferencial Profesor: Daniel Rodríguez

May 2015

Abstract

This paper describes the technologies needed to build a low speed, short range and low transmission power WPAN, intended to do the telemetry of many communication nodes, distributed in a group of cabins in a small holiday site, so as to collect these measurements into a central node, which will receive this information from the End-Devices and deliver them to a measurement service located on the Internet.

The wireless transmission technology used in this work is the protocol miwi from microchip, which is based on IEEE 802.15.4 protocol, and operates in the frequencies spectrum of 2.4 GHz

The logic control system is based on the chip 18F27J53 member of 8 bit PIC18 microchip microcontroller's family, which is programed using C language under the development environment MPLAB X, using the protocol stack indicated above.

As a tool for programming and debugging we have chosen the ICD3 of the same provider, in order to facilitate the process of software debugging and minimize the TTM (Time to Market), a matter of vital importance in any technological development today.

In short, bringing together all these elements and technologies, a simple, economical, low power consumption, and highly flexible design is achieved to implement the solution to the problem of automatically control the occupancy of the cabins, leaving the owner more time for other tasks, minimizing the losses caused by the elusion of payments from the local administrator.

The Web application that receives and processes this information is beyond the scope of this work, and will only be presented a version of simple test that shows that the information sent from the sensors is received and stored.

Key words: telemetry, WPAN, Personal area network

Glosario

WPAN : Wireless Personal Area Network

IrDa: Infrared Data Association

Wifi: Nombre comercial de wi-fi alliance

GSM: Global system for Mobile

GPRS: General Packet Radio service

PIR: Pasive Infrared Sensor

LDR: Light Dependant Resistor

LCD: Liquid crystal display

MCU: MicroControllerUnit

SPI: Serial Periferal Interface

SCK: Source clock (SPI)

MOSI: Master Out slave In (SPI)

MISO: Master In Slave Out (SPI)

RTCC: Real Time Clock and Calendar

EEPROM: Electrically Erasable Programmable Read-Only Memory

WDT: Watch Dog Timer

MSSP: Master Synchronous Serial Port

MAC: Media Access Control

CRC: Cyclic Redundancy Check

PA: Power Amplifier

LNA: Low Noiser Amplifier

CSMA-CA: Carrier sense multiple access with collision avoidance

FCS: Frame Check Secuence

IDE: Integrated Development environment

WDS: Wireless Developers Studio

GPIO: General Purpose Input Output

MCLR: Master Clear

Índice general

Resumen.....	i
Abstract	ii
Glosario.....	iii
Índice de figuras.....	vi
Índice de Tablas	vii
1. Introducción.....	1
2. Estado del arte	2
2.1. Requerimientos para el sistema	2
2.2. Tecnologías y productos actuales	3
2.3. Análisis y selección de Tecnologías	4
Bluetooth	4
IrDa.....	4
XBee.....	4
Wifi.....	4
GSM / GPRS	5
Miwi	5
Zigbee.....	5
3. Análisis y Diseño.....	7
3.1. Diseño del End-Device	9
3.1.1. Hardware.....	9
3.1.2. Software	10
3.2. Diseño del Base-Device.....	11
3.2.1. Hardware.....	11
3.2.2. Software	13
3.3. Elección de Componentes.....	14
3.3.1. Microcontrolador	14
3.3.2. Módulo Ethernet	16
3.3.3. Módulo Wireless.....	17
3.3.4. LCD	18
3.3.5. Sensores PIR y LDR	19
4. Implementación del sistema	20
4.1. Ambiente de trabajo.....	20

4.1.1.	ICD3.....	20
4.1.2.	MPLABX.....	21
4.1.3.	C18.....	21
4.1.4.	WDS y Zena.....	21
4.2.	Implementación del End-Device	22
4.2.1.	MCU	22
4.2.2.	LCD	24
4.2.3.	MIWI	26
4.2.4.	WhatchDog Timer	32
4.2.5.	Timer0.....	34
4.2.6.	Linker.....	37
4.2.7.	Sensores	39
4.2.8.	Sensores alternativos.....	40
4.3.	Implementación de la Base.....	41
4.3.1.	MCU	41
4.3.2.	MIWI	45
4.4.	Desarrollo del WebServer de Prueba.....	46
4.5.	Prueba del sistema y Mediciones de campo	47
5.	Conclusiones y Trabajo Futuro.....	53
6.	Referencias	54

Índice de figuras

Figura 2-1: Mapa de tecnologías de comunicación inalámbrica según alcance y velocidad de transmisión	3
Figura 3-1: Diagrama de conexión general	8
Figura 3-2: Diagrama de diseño de hardware del End-Device	9
Figura 3-3: Diagrama de diseño de hardware del Base-Device	11
Figura 3-4: PIC 18F27J53	15
Figura 3-5: Módulo ENC28J60	16
Figura 3-6: CPU ENC28J60	16
Figura 3-7: Módulo MRF24J40MC	17
Figura 3-8: EEPROM 25AA02E48	17
Figura 3-9: CPU MRF24J40	17
Figura 3-10: LCD	18
Figura 3-11: Módulo PIR	19
Figura 3-12: LDR	19
Figura 4-1: herramienta de programación y depuración ICD3	20
Figura 4-2: MPLABX	21
Figura 4-3: WDS y Zena	21
Figura 4-4: End-Device	22
Figura 4-5: Flujo de inicialización del End-Device	23
Figura 4-6: Flujo de inicialización del LCD	25
Figura 4-7: Flujo de inicialización Miwi en End-Device	26
Figura 4-8: Diagrama de flujo aplicación End-Device	28
Figura 4-9: Diagrama de flujo control EEPROM	29
Figura 4-10: Diagrama de bloques Timer 0	36
Figura 4-11: Sensores alternativos	40
Figura 4-12: Base-Device	41
Figura 4-13: Flujo de inicialización Base-Device	42
Figura 4-14: Diagrama de flujo aplicación Base-Device	43
Figura 4-15: Flujo de inicialización Miwi en Base-device	45
Figura 4-16: Diagrama del ambiente de testing	47
Figura 4-17: Gráfico de mediciones de ocupación obtenidas del sensor PIR	48
Figura 4-18: Gráfico de mediciones obtenidas del sensor LDR	49
Figura 4-19: Correlación detección PIR y LDR	49
Figura 4-20: Captura de tráfico de red entre Base-Device y Web Server	50
Figura 4-21: Señales MISO (en amarillo) y SCK (en azul) del bus SPI	51
Figura 4-22: Señales MOSI y SCK del bus SPI	51
Figura 4-23: Señal de salida del módulo PIR	52

Índice de Tablas

Tabla 2-1: Comparación de tecnologías de WPAN..... 6

Tabla 3-1: Requerimientos del software para el End-Device..... 10

Tabla 3-2: Requerimientos del software para el Base-Device..... 13

Tabla 4-1: Protocolo de comunicación Base-Device - End-Device 30

1. Introducción

La presente memoria busca abordar una problemática particular detectada a través de entrevistas con diversos dueños de cabañas, donde expresan el deseo de poder alejarse de sus unidades con la confianza de no perder la capacidad de arriendo ni de verse engañados por sus administradores quienes en ocasiones arriendan dichas unidades a un menor valor del indicado, sin entregar la boleta del servicio y sin dejar registros, lo que se traduce en la pérdida del dueño de este dinero y del robo por parte de su administrador.

Con el problema en mente, definiremos los objetivos necesarios para la construcción de un sistema de software y hardware que permita cumplir con los propósitos de este diseño los cuales se enumeran a continuación:

- Controlar la ocupación en las dependencias disponibles, evitando el mal uso de las mismas.
- Identificar situaciones anómalas, como la falta de reporte desde las cabañas o la recepción de información contradictoria de los diversos sensores.
- Mejorar la capacidad de gestión de sitios para su dueño, a diversas zonas geográficas, remotamente distantes.
- Liberar al dueño de las cabañas para la realización de otras actividades.

Con estos objetivos y considerando que la distancia entre cada cabaña es de entre 5 y 40 metros, estudiaremos las alternativas existentes en el mercado actual de forma de buscar su aplicación o haremos el desarrollo de algún sistema apropiado considerando las alternativas tecnológicas existentes.

2. Estado del arte

En este capítulo nos detendremos a explicar las características o requerimientos para el Hardware / Software del sistema y compararemos estos contra las tecnología y productos actualmente disponibles para justificar la elección del mejor camino a seguir para su desarrollo o implementación.

2.1. Requerimientos para el sistema

- Simple de instalar y operar.
- Mínima intervención de la infraestructura actual.
- Muy bajo costo de instalación y mantención.
- Muy bajo consumo de energía para ser alimentado por baterías (Recolección de datos con o sin energía eléctrica) en los End-Device.
- Comunicación multipunto con ruteo para llegar a zonas alejadas del punto central de control, con un mínimo de 10 saltos.
- Capacidad de comunicación fuera de línea visual, de al menos 30 metros.
- Mínimo de 50 End-Devices con seguridad en el protocolo.
- Comunicación con servidor web en Internet para funcionamiento como servicio.

2.2. Tecnologías y productos actuales

Para cubrir la necesidad de baja intervención de los ambientes donde se instale el sistema y bajos costos, será necesario considerar alguna de las tecnologías de transmisión inalámbrica existentes hoy en día. Se debe considerar además que muchas de las cabañas donde este sistema se instale no tienen un buen sistema wifi en operación, o no tienen la cobertura adecuada.

Las posibles alternativas de tecnologías y productos presentes en la actualidad son los indicados abajo:

- Bluetooth
- IrDa
- XBee
- Wifi
- GSM / GPRS
- MiWi
- Zigbee

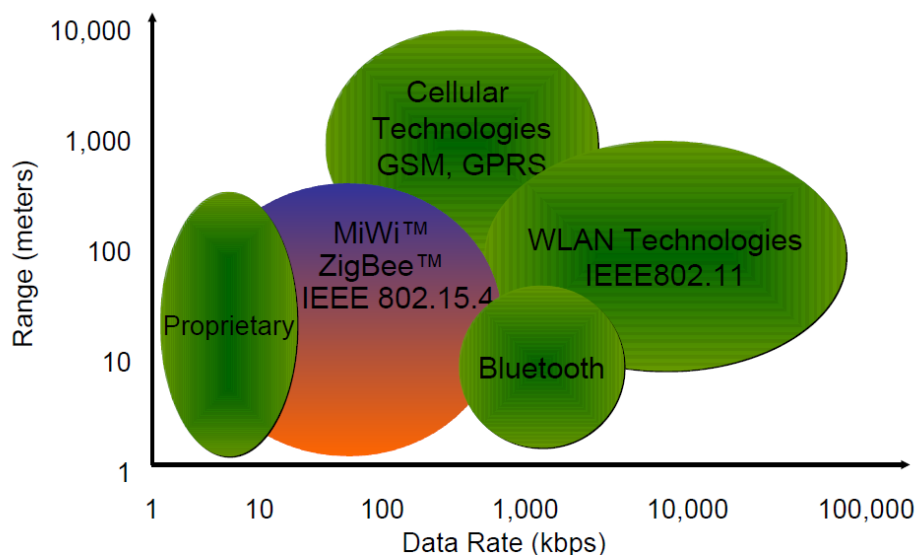


Figura 2-1: Mapa de tecnologías de comunicación inalámbrica según alcance y velocidad de transmisión

En la Figura 2-1 se muestran las diferentes alternativas según el alcance y velocidad de transmisión de datos. Dado nuestros requerimientos de distancia y baja tasa de transmisión a priori parecen más adecuadas las tecnologías relativas al protocolo 802.15.4.

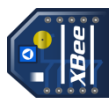
2.3. Análisis y selección de Tecnologías

A continuación se hará el análisis de cada una de las posibles tecnologías y productos para nuestro proyecto para definir el más adecuado.

Bluetooth:  **Bluetooth®** Si bien existen dispositivos Clase 1 con hasta 15dBm a 20dBm de potencia, este protocolo está orientado hacia grandes tasas de transferencia de datos, sin tener capacidades de crear redes mesh con el objetivo de alcanzar dispositivos fuera del alcance directo (routing de paquetes). Además se tiene un límite de 7 conexiones Slave por cada Master, por lo que claramente no es adecuado para nuestro requerimiento, donde el número de cabañas a ser monitoreado es de un mínimo de 50, con una distancia de 30 metros sin línea visual directa.



IrDa: Tecnologías de transmisión infrarroja, adaptada a cortas distancias de transmisión, con línea de vista entre transmisor y receptor, no adecuado para nuestro propósito.



XBee: Producto comercial basado en IEEE 802.15.4, pensado para agregar rápidamente características de comunicación inalámbrica a cualquier equipo mediante el control serial UART, el que puede ser provisto por un PC, Microcontrolador, o cualquier equipo similar. Tiene excelentes prestaciones para aplicaciones simples, y posee además la capacidad de agregar routers con conexión a internet a la WPAN, permitiendo incluso la configuración y control mediante una página provista por el fabricante.



Wifi : Protocolo basado en IEEE 802.11 con alta velocidad de transferencia, capacidad de múltiples tipos de redes y gran cantidad de equipos comerciales a disposición. Sus grandes prestaciones se convierten en desventajas para este proyecto dado que su compleja arquitectura requiere de varios dispositivos como repetidores y routers para lograr extender el alcance de la red. Por otra parte su costo es elevado, así como el consumo de energía. Por lo tanto, no será considerado como una opción válida para este trabajo.



GSM / GPRS: Servicio de envío de datos por la red móvil con un alcance que depende básicamente de la cobertura radial del servicio del proveedor y una tasa de transferencia de datos de entre 20 y 100 kbps, suficientes para nuestra aplicación. Si bien existen módulos usables por microcontroladores y aplicaciones comerciales, éstas son de elevado costo y consumo de energía.

Además es importante notar que existe un cobro asociado por el uso de la red del portador, que depende del consumo de ancho de banda y tamaño de la información intercambiada.

Por lo anterior, este tipo de comunicación es más apropiado para zonas aisladas o de difícil acceso, quedando descartado para nuestro proyecto, en relación a la creación de una red PAN.



Miwi: Protocolo asociado a un producto comercial de Microchip basado en IEEE 802.15.4, con dos diferentes versiones para aplicaciones P2P y para múltiples nodos, pudiendo ser usado en muy pequeños MCU de 8 bits, hasta las MCU de gama superior. Capacidades de formar redes P2P, estrella, malla.

Cuenta con Transceivers intercambiables fácilmente para pasar de 2.4GHz a la banda de 900MHz, con manejo automático de reenvíos (Handshaked), ruteo, muy bajo consumo de energía, y el almacenamiento de información desde y hacia los nodos en modo Deep Sleep.

Además de lo anterior, este protocolo tiene un bajo footprint y consumo de RAM en la MCU, y cuenta con un ambiente de desarrollo provisto por Microchip, donde se incluye el stack de programación de forma gratuita para uso en MCU de su marca, y el ambiente de desarrollo y compilador. Importante es notar que el mismo hardware de Microchip se puede usar con el stack de Zigbee, pero se debe pagar un royalty por su uso.



Zigbee: Protocolo estándar basado en IEEE 802.15.4 de características muy similares a Miwi (Miwi nace de la definición de Zigbee). Permite topologías de estrella, árbol y malla. Su uso, al igual que en Miwi, requiere de pocos componentes, pero su footprint es considerablemente mayor, requiriéndose MCU de gama media o alta para su implementación. También tiene la capacidad de intercambiar transceivers para pasar de la banda de 2.4GHz a 900MHz y de manejar paquetes de nodos lejanos al coordinador de la red. Su desarrollo en aplicaciones comerciales requiere el pago de un Royalty por el uso del protocolo y además de

esto es necesaria la certificación de los productos adheridos a este estándar, lo que supone nuevos costos.

La Tabla 2-1 muestra la comparación de las principales tecnologías viables para la implementación de este proyecto.

Tabla 2-1: Comparación de tecnologías de WPAN

	Alcance indoor (mts)	Alcance outdoor (mts)	Num de saltos	Numero de End Devices	Frecuencia	BW (kbps)	Secure	Tx power (mw)	Consumo Tx (mA)	Consumo Rx (mA)	Consumo Reposo (uA)	Footprint kbyte (Coord / EndDevice)
Zigbee	20	1500	>1000	65000	900MHz / 2.4 GHz	20 - 250	128 bit	1 -100	N/D	N/D	N/D	40 - 21
Xbee-pro	90	1600		65000	2.4 GHz	250	128 bit	63	250	55	10	N/A
Miwi-pro	20	2000	64	8000	900MHz / 2.4 GHz	250	128 bit	0 - 100	120	25	12	25 - 15

En términos de aplicaciones comerciales desarrolladas para el propósito de la presente memoria, podemos indicar que según la búsqueda efectuada no existe ninguna iniciativa en esta línea en ningún lugar del mundo, encontrándose diferentes productos orientados al control de la iluminación, alarmas, y telemetría, donde el estándar de facto usado es Zigbee, principalmente por la interoperabilidad de componentes desarrollados por distintos fabricantes.

Considerando lo expuesto en este capítulo, para este proyecto de memoria se utilizará la tecnología Miwi de Microchip, por ser más económica, fácil de implementar e igualmente potente que Zigbee para la aplicación descrita, sin requerirse compatibilidad con equipos de terceros.

3. Análisis y Diseño

Para el diseño del sistema de detección de ocupación de cabañas se deben considerar algunas condiciones importantes para el éxito del sistema.

- La red de transmisión inalámbrica debe ser autónoma y no depender de otra infraestructura.
- El alcance de transmisión inalámbrica a considerar es desde el Base-Device hacia los End-Device debe ser de al menos 30 metros, estando la base ubicada en la oficina de administración y los End-Devices en cada una de las cabañas.
- Debe ser posible alcanzar cabañas lejanas usando las cabañas intermedias como puente (capacidad de ruteo).
- Los End-Devices deben ser alimentados con pilas, las que deben tener una duración aceptable.
- La base se debe conectar a Internet mediante cable, conector RJ45 y debe ser alimentado con una fuente fija para soportar el consumo generado por la transmisión Ethernet.
- El costo de los dispositivos debe ser lo más bajo posible para facilitar modelos de negocio sin venta, y pago solo por el servicio.
- Debe ser simple de instalar y mantener.

Considerando lo anterior, se establecerá una red PAN (Red de área personal) usando el protocolo Miwi de Microchip, siendo la Base la creadora de la red.

Los dispositivos End-Devices a ser instalados en las cabañas buscarán conectarse con una red existente (creada por el Base-Device) para poder reportar los eventos detectados por sus sensores.

Los dispositivos End-Device que estén fuera del alcance directo de la base podrán unirse a la red mediante otro End-Device que esté al alcance de su señal, siendo esto una propiedad del protocolo Miwi, sin requerir de alguna programación especial para que esto ocurra.

La Base reportará permanentemente el estado de la red de sensores hacia un WebServer quien registrará la información enviada en una base de datos donde podrá aplicar algoritmos conducentes a la detección de situaciones anómalas y a la detección de los períodos en los cuales registra ocupación de arrendatarios, descartando las detecciones producidas por personal de aseo u otras situaciones similares.

En la presente memoria no se desarrollarán dichos algoritmos, bastando el registro de la información en el servidor.

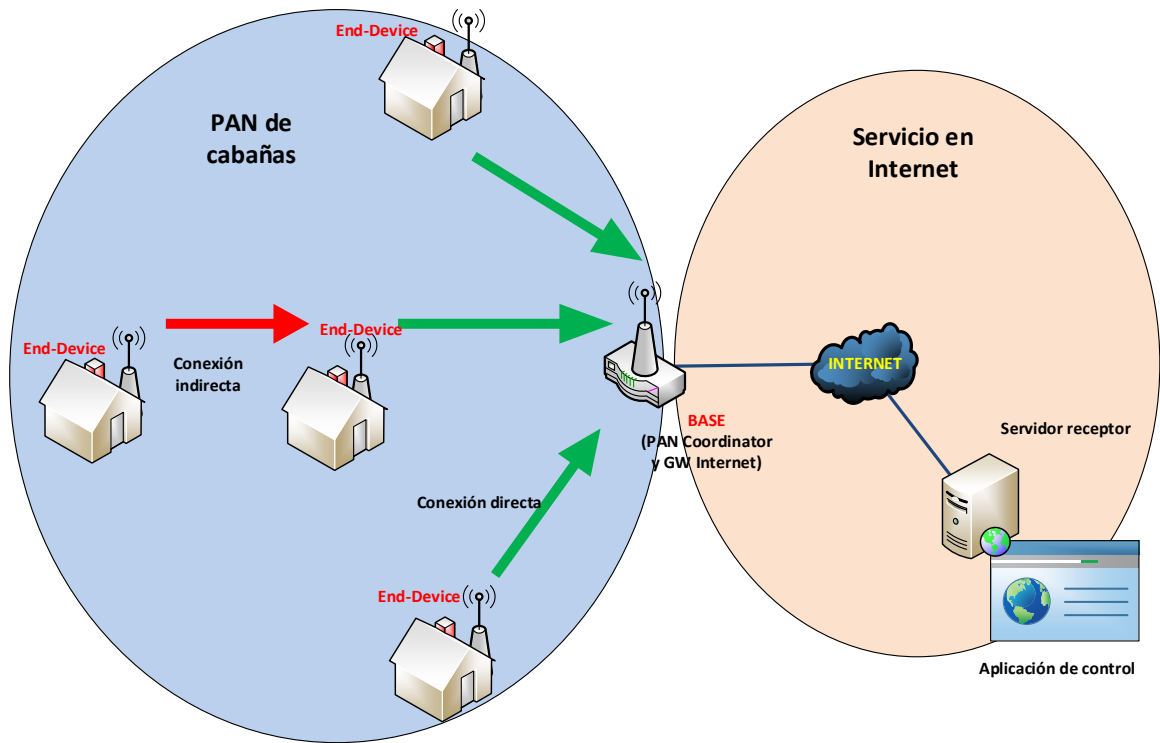


Figura 3-1: Diagrama de conexión general

La Figura 3-1 muestra el diagrama de conexión entre los End-Devices existentes en la PAN, con la base y el webservice que registrará toda la información enviada.

En el diagrama se identifican mediante flechas verdes las situaciones en que un End-Device está dentro del alcance del Base-Device y los que están fuera del alcance directo (flechas en rojo), pero dentro del alcance de un End-Device que actúa como router, extendiendo la red al mismo.

3.1. Diseño del End-Device

3.1.1. Hardware

Para el dispositivo que se ubicará en la cabaña es necesario contar con los módulos:

- Microcontrolador
- Wireless
- Sensor PIR y LDR
- LCD
- Fuente de poder (mediante un regulador MCP1702 de 3.3v, y 3 pilas AA)

Esquemáticamente la interconexión es la mostrada en la Figura 3-2

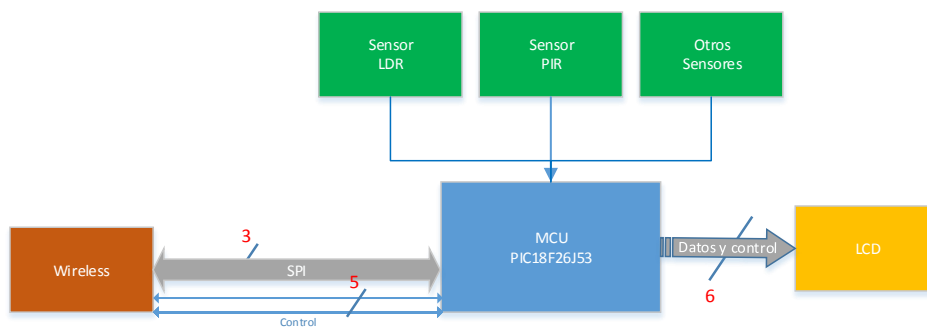


Figura 3-2: Diagrama de diseño de hardware del End-Device

Los buses de conexión y control desde el MCU son los siguientes:

- Conexión con PIR
 - Una entrada digital
 - Una salida digital (led de testing)
- Conexión con LDR
 - Una entrada análoga
- Conexión con LCD
 - 4 salidas digitales para datos
 - Salida digital para RS
 - Salida digital para E
- Conexión con Wireless
 - Salida digital para SCK
 - Salida digital para MOSI
 - Entrada digital para MISO
 - Physical_Chip_Select
 - RF_Interrupcion
 - Physical_Wake
 - RF_EEPROM_Chip_Select
 - Physical_Reset

3.1.2. Software

El End-device debe tener la capacidad de conectarse con la red creada por la base e intercambiar información con ésta.

Inicialmente el End-device debe obtener la fecha y hora desde la base para poder configurar su referencia interna de hora RTCC quedando sincronizado con la base. El objetivo principal de esto es el almacenamiento de información de las detecciones monitoreadas en caso que la base no esté presente por alguna razón. Dicho almacenamiento de información debe hacerse con la fecha y hora de modo de poder enviarlo posteriormente a la base y que el servidor web pueda unir estas mediciones perdidas junto al resto de las mediciones normalmente recibidas en la misma línea de tiempo.

En un ciclo de 10 minutos el sensor enviará información de las detecciones hechas, que son equivalentes a la cantidad de disparos del sensor detectados dentro de ese tiempo o cambios de estados de la entrada. Además se envía la medida de la luz medida, tomada por un conversor A/D sobre el sensor LDR.

Como se indicó inicialmente se usará la memoria EEPROM del módulo Miwi (25AA02E48 EUI de 256 bytes) para almacenar la información que no se pueda enviar a la base, por lo que es requisito que el End-Device pueda controlar la escritura y lectura de esta memoria.

Existen además mensajes que se inician desde la base y que deben ser procesados por el End-Device como son el envío del timestamp una vez por día y el requerimiento de información del estado del dispositivo.

El resumen de los requerimientos generales para el software del End-Device son los presentados en la Tabla 3-1.

Tabla 3-1: Requerimientos del software para el End-Device

N°	Función	Descripción
1	Buscar red Miwi y conectarse	Mediante Miwi debe buscar redes presentes y unirse
2	Obtener DateTime para programar su RTCC	Mediante Miwi debe obtener DateTime de la base y configurarse con este
3	Recibir y procesar mensajes desde la base	Base enviará mensajes con DateTime una vez por día y puede preguntar el estatus
4	Contar cantidad de cambios del sensor PIR	Obtener una referencia de cantidad de uso de la dependencia
5	Enviar la lectura de los sensores	Además de los sensores PIR y LDR, se pueden agregar otros
6	Guardar los datos en EEPROM en caso de falla	En caso de falla en la transmisión hacia la base debe guardar los datos.
7	Administrar la información guardada en EEPROM	Guardar y eliminar información según sea necesario
8	Retransmitir información guardada	Retransmite el contenido de la EEPROM
9	Probar sensor y enviar información a Base	Se prueba mediante el encendido de un led

3.2. Diseño del Base-Device

3.2.1. Hardware

Para el dispositivo que se ubicará en la administración es necesario contar con los módulos:

- Microcontrolador
- Wireless
- Ethernet
- Sensor PIR
- LCD
- Fuente de poder (mediante un regulador MCP1702 de 3.3v, y 3 pilas AA)

Esquemáticamente la interconexión es la mostrada en la Figura 3-3.

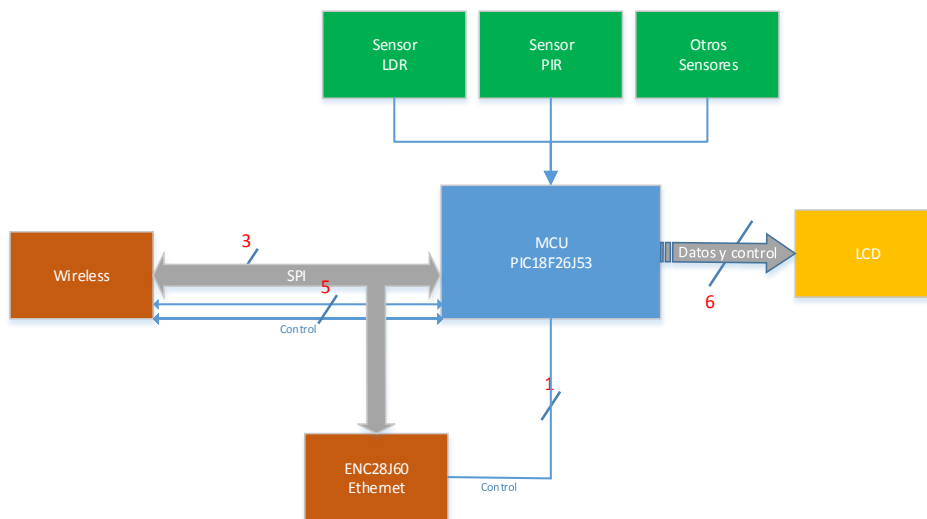


Figura 3-3: Diagrama de diseño de hardware del Base-Device

Los buses de conexión y control desde el MCU son los siguientes:

- Conexión con PIR
 - Una entrada digital
 - Una salida digital (led de testing)
- Conexión con LDR
 - Una entrada analógica de conversor A/D
- Conexión con LCD
 - 4 salidas digitales para datos
 - Salida digital para RS
 - Salida digital para E

- Conexión con Wireless
 - Salida digital para SCK
 - Salida digital para MOSI
 - Entrada digital para MISO
 - Physical_Chip_Select
 - RF_Interrupcion
 - Physical_Wake
 - RF_EEPROM_Chip_Select
 - Physical_Reset
- Conexión con Ethernet
 - Salida digital para SCK (mismo bus con Wireless)
 - Salida digital para MOSI (mismo bus con Wireless)
 - Entrada digital para MISO (mismo bus con Wireless)
 - Salida digital para ENC_Chip_Select
 - Entrada análoga para conversor A/D (Fuente de entropía)

3.2.2. Software

Inicialmente la Base configura sus registros y sus periféricos, y antes de poder crear la red Wireless debe lograr establecer la comunicación con el servidor Web desde el que obtiene la fecha y hora para configurar su referencia de tiempo interna RTCC. Esto es fundamental ya que la base deberá entregar a cada End-Device la misma referencia de tiempo.

En un ciclo de 10 minutos la base recolecta la información proveniente desde los End-Devices y la agrega a su estructura de control para transmitirlo al servidor web una vez que ha recibido la información desde todos los sensores. En caso de no haber recibido todas las mediciones, agregará un valor arbitrario en el End-Device no recibido para señalar esta condición.

Por diseño, las lecturas válidas desde los sensores van desde cero a doscientos cincuenta (0-250). Además de estos valores la base podría no recibir nada en el evento de la pérdida de comunicación con un End-Device, en cuyo caso transmitirá hacia el web server el valor 251.

Una vez sincronizado con el servidor web, la base crea la red miwi para recibir las conexiones desde los End-Devices.

También se usará la memoria EEPROM del módulo Miwi para almacenar la información que no se pueda enviar al servidor web.

El resumen de los requerimientos generales para el software para el Base-Device son los indicados en la Tabla 3-2.

Tabla 3-2: Requerimientos del software para el Base-Device

N°	Función	Descripción
1	Obtener DateTime desde servidor web para programar su RTCC	Mediante Ethernet debe obtener el datetime desde el server
2	Crear red miwi	Mediante Miwi debe obtener DateTime de la base y configurarse con este
3	Recibir y procesar mensajes desde End-Devices	Base debe recibir información desde los End-Devices y procesarlos para guardarlos en memoria a la espera de que todos completen sus transmisiones
4	Enviar información recolectada al web server	Mediante módulo ENC28j60 enviar datos hacia el webserver
5	Contar cantidad de cambios del sensor PIR	Obtener una referencia de cantidad de uso de la dependencia
6	Medir la cantidad de luz del sensor LDR	Obtener una lectura de la intensidad de la luz
7	Guardar los datos en EEPROM en caso de falla	En caso de falla en la transmisión hacia el webserver guardar los datos.
8	Administrar la información guardada en EEPROM	Guardar y eliminar información según sea necesario
9	Retransmitir información guardada	Retransmite el contenido de la EEPROM
10	Probar sensores	Se prueba mediante el encendido de un led

3.3. Elección de Componentes

3.3.1. Microcontrolador

Dado que para nuestro propósito se usarán los sistemas Miwi de Microchip, se seleccionará también el microcontrolador de este fabricante de forma de contar con un solo ambiente de trabajo para la programación y debug del sistema.

Además y dado que el prototipo se montara sobre un protoboard, elegiremos un microcontrolador en encapsulado SPDIP.

Otros requisitos son:

- Capacidad suficiente de RAM que soporte el stack Miwi y Ethernet (Base)
- Capacidad suficiente de memoria de programa para el desarrollo
- RTCC
- Dos conversores A/D para usar como fuente de entropía para Ethernet y sensor LDR
- Entradas y salidas digitales para todos los periféricos
- SPI
- Pines de Interrupciones reprogramables
- Timers
- Mínima necesidad de componentes adicionales

Con estas características encontramos algunas familias de MCU adecuadas para el propósito, siendo escogido el PIC18F27J53 cuyo diagrama de pines se muestra en la Figura 3-4.

28-Pin SPDIP/SOIC/SSOP

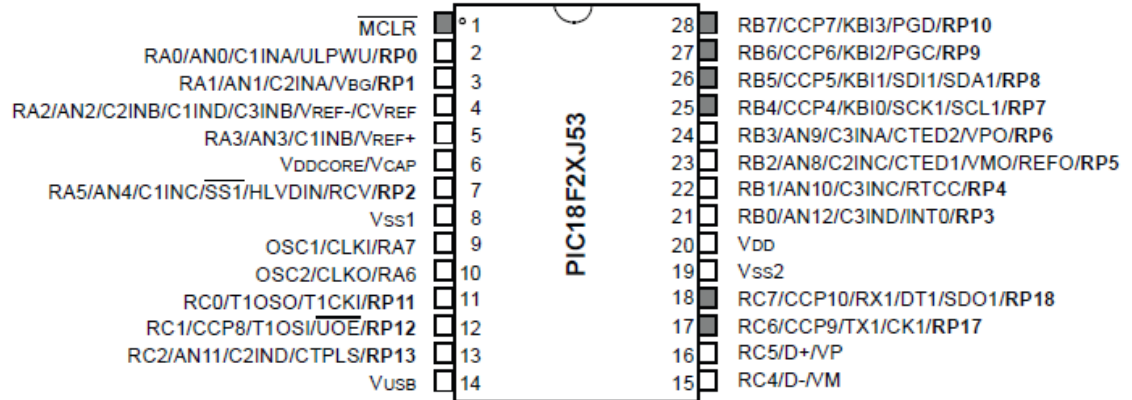


Figura 3-4: PIC 18F27J53

Sus principales características relevantes al proyecto son:

- Múltiples modos de Sleep con consumos de hasta 13nA
- Doble oscilador interno de frecuencia ajustable
- WDT desde 4ms hasta 131s
- Operación desde 2.0V a 3.6V
- Permite el mapeo de pines I/O
- RTCC
- Doble puerto MSSP para I2C y SPI
- 28 pines, 128 kbytes program, 3.8kbytes RAM.

Es importante notar que aunque este MCU es adecuado para el Proyecto, en caso de necesidad, se puede escalar con otros controladores de 8 y de 16 bits con mínimos cambios sobre el software desarrollado.

3.3.2. Módulo Ethernet

Para el módulo Ethernet se elige el Olimex basado en el ENC28j60 de microchip, dado que cuenta con todos los componentes necesarios para comenzar a trabajar directamente con el módulo usando el puerto serial con el protocolo SPI, facilitando el desarrollo y minimizando el tiempo necesario para agregar comunicación Ethernet a una MCU.



Figura 3-5: Módulo ENC28J60

Las principales características de este módulo son:

- No requiere componentes externos
- Compatible con redes 10/100/1000 BaseT
- MAC y capa física 10Base-T incorporados
- Soporta un puerto 10Base-T con detección de polaridad
- Soporta modos Full y Half para dúplex
- Retransmisión automática ante colisiones
- Generación de Padding y CRC
- Interfaz SPI de hasta 20Mhz
- Memoria de 8kbyte para transmisión y recepción
- Operación desde 3.1V a 3.6V

28-Pin SPDIP, SSOP, SOIC

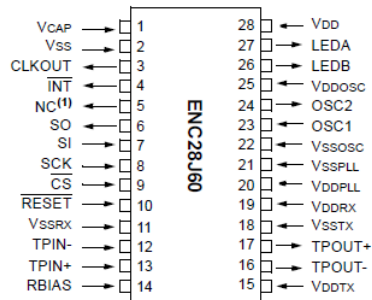


Figura 3-6: CPU ENC28J60

3.3.3. Módulo Wireless

Para el módulo Wireless se escoge el MRF24J40MC de microchip dado que éste utiliza el transceiver MRF24J40, con un amplificador de señal PA/LNA, antena externa y una memoria EEPROM que contiene el MAC Address entregado por la fábrica directamente.



Figura 3-7: Módulo MRF24J40MC

Las principales características de este módulo son:

- Potencia de transmisión de hasta 18,5 dbm
- Control mediante SPI, con interrupción
- Trae integrados el cristal, regulador de tensión, PA/LNA y componentes secundarios.
- Conector de antena externa
- Certificado FCC / IC / ETSI
- Banda ISM 2,405 GHz hasta 2,475 GHz
- -108 dBm de sensibilidad
- Mecanismo CSMA-CA por hardware, Ack y chequeo FCS automáticos
- Capacidad de retransmisión de paquetes automático
- Seguridad AES-128
- Memoria EEPROM 25AA02E48 EUI de 256 bytes incorporado.

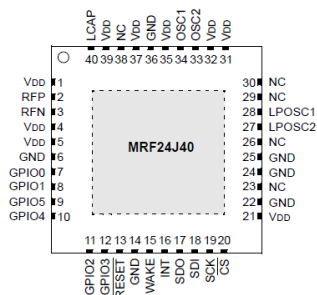


Figura 3-9: CPU MRF24J40

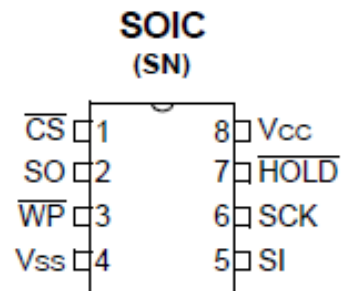


Figura 3-8: EEPROM 25AA02E48

3.3.4.LCD

Principalmente para propósitos de debug y de entender el estado del sistema de monitoreo se incorpora un módulo LCD WH2002A basado en el circuito ST7066, compatible con HD44780 de Hitachi.

Las principales características del módulo son:

- 2 líneas de 20 caracteres cada una
- Alimentación de 3V a 5V
- Control por bus de 8 o 4 bits



Figura 3-10: LCD

3.3.5.Sensores PIR y LDR

Para el sensor de movimiento que se usará en las estaciones End-Device se elige el módulo PIR de Parallax.

Sus principales características son:

- Detección de hasta 10 metros de distancia
- Salida por un único pin
- Selección de salida en pulso simple o continuo por jumper
- Operación desde 3.3V a 5V



Figura 3-11: Módulo PIR

El sensor LDR es simplemente este componente, conectado mediante un divisor de tensión al pin análogo del MCU de modo que este pueda hacer la conversión A/D



Figura 3-12: LDR

4. Implementación del sistema

4.1. Ambiente de trabajo

En la implementación del sistema fueron usados los componentes ya descritos unidos por el desarrollo de la aplicación que debe cumplir con los requisitos antes establecidos.

El prototipo fue montado en un Protoboard para facilitar el proceso de cambios en el hardware y construcción del mismo, y fueron usadas las herramientas de Microchip para el desarrollo y debugging.

4.1.1. ICD3

Sistema para la programación en placa del circuito (ICSP) y debug del mismo



Figura 4-1: herramienta de programación y depuración ICD3

4.1.2.MPLABX

IDE para el desarrollo de software que integra al ICD3 como herramienta y une los compiladores y ensambladores

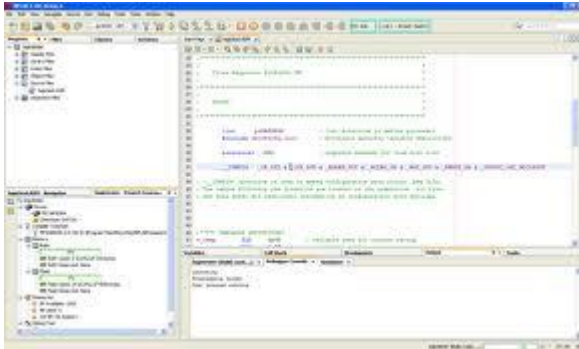


Figura 4-2: MPLABX

4.1.3.C18

Compilador de C usado para el proyecto y configurado dentro del entorno de trabajo MPLABX

4.1.4.WDS y Zena

Para lograr revisar la información que se trafica por vía aérea en miwi se usó el Wireless development studio y el hardware Zena que permite funcionar como un sniffer inalámbrico.

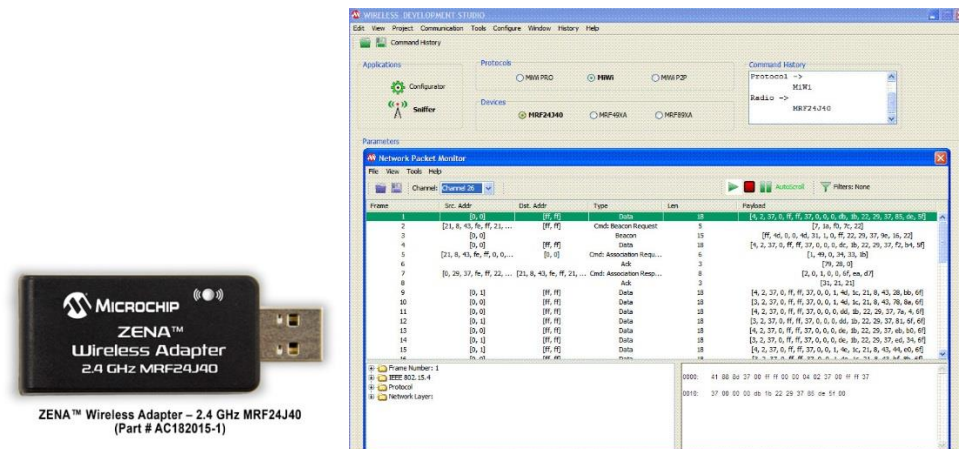


Figura 4-3: WDS y Zena

4.2. Implementación del End-Device

La Figura 4-4 muestra el prototipo del End-device con todos sus componentes y módulos cuya funcionalidad dentro del proyecto será descrita a continuación.

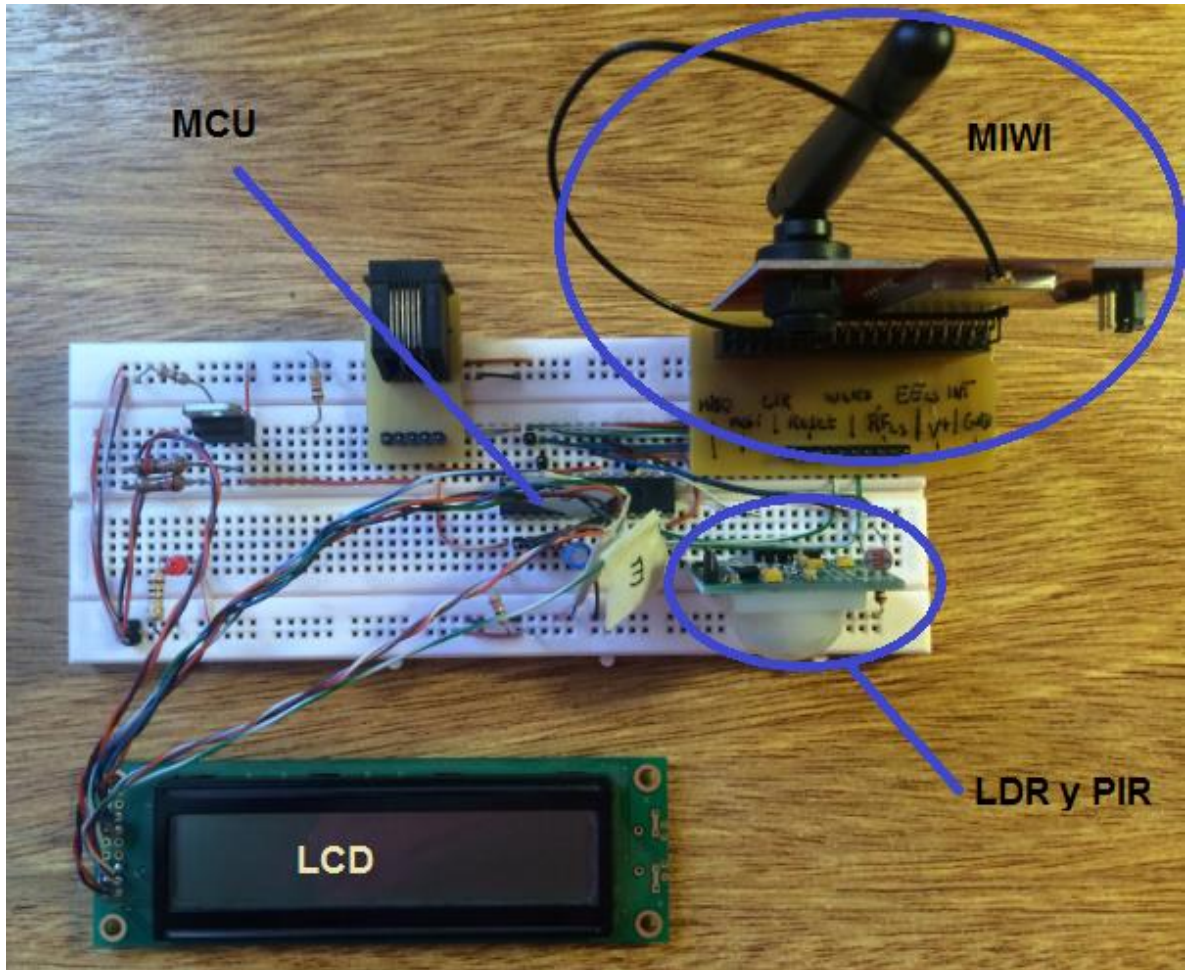


Figura 4-4: End-Device

4.2.1.MCU

Es la responsable de la ejecución del código programado. Lo primero que se debe hacer al iniciar el sistema, es configurar todas las opciones adecuadas dentro de la MCU, como los redireccionamientos de pines, frecuencia del oscilador, definición de los puertos de entrada y salida, definición de entradas análogas o digitales, deshabilitación de módulos no usados en la MCU, configuración del subsistema SPI y configuración de registros.

La Figura 4-5 muestra dichas acciones de inicialización del sistema

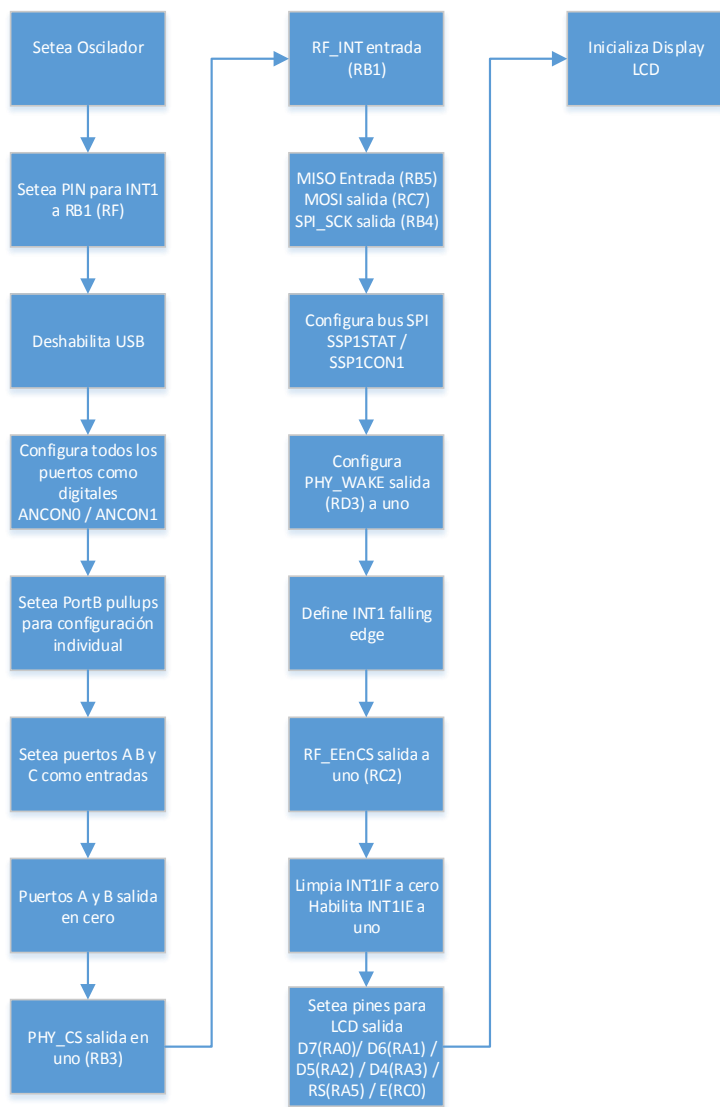


Figura 4-5: Flujo de inicialización del End-Device

4.2.2.LCD

El desarrollo del End-Device se inicia con el manejo y control del LCD que será de gran ayuda para poder entender el estado del circuito, entregando mensajes programados y valores de variables que nos indiquen si el flujo del programa es el esperado.

Para lograr este hito, se estudia la forma de comunicación del LCD, el que cuenta con las entradas RS y E además de 4 u 8 líneas de datos.

En nuestro diseño, para ahorrar en el uso de pines GPIO, se implementó la modalidad de comunicación en 4 bits, donde la información es transmitida en dos mitades. La entrada RS sirve para selección del registro a ser usado, pudiendo ser estas instrucciones o datos. R/W sirve para seleccionar una operación de lectura o escritura. En nuestro diseño esta entrada del LCD estará permanentemente conectada a tierra. La entrada E sirve para iniciar una transmisión de datos de lectura o escritura.

Para inicializar el LCD es necesario enviar una secuencia de instrucciones dentro de parámetros de tiempo establecidos, los que se muestran en la Figura 4-6 de forma gráfica y además el código de programación asociado.

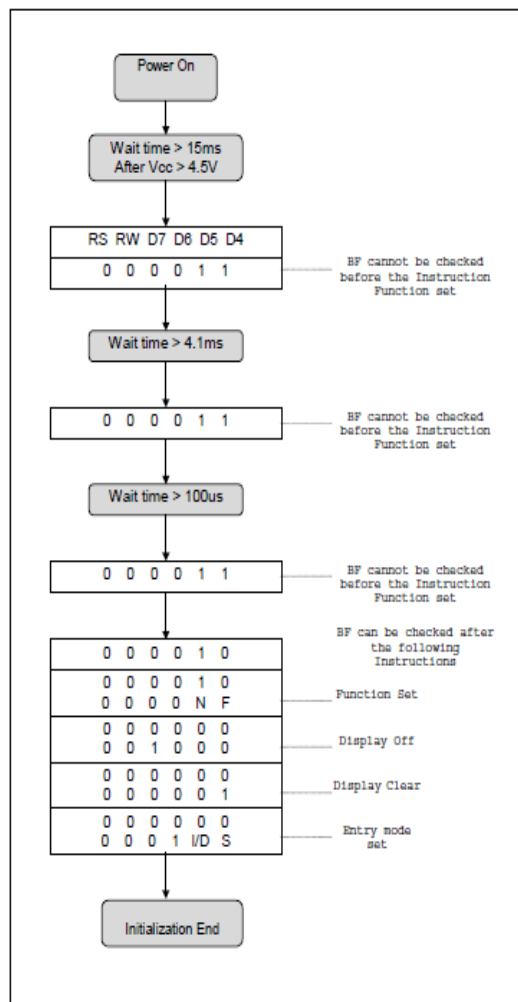


Figura 4-6: Flujo de inicialización del LCD

Una vez inicializado el LCD es posible enviar instrucciones para el despliegue de mensajes, métodos que fueron desarrollados en el proyecto como funciones dentro del fichero lcd.c

Código asociado

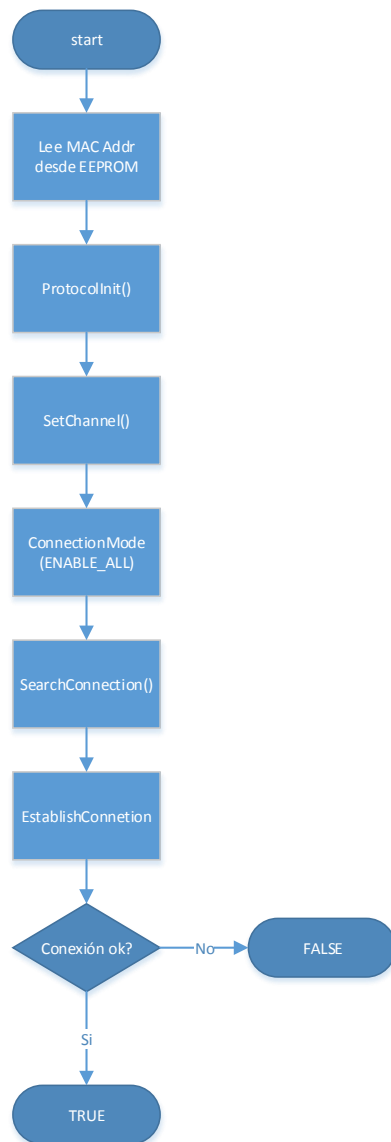
void display_init()

```

{
    RS = 0;
    E = 0;
    DelayMs(15);
    set_Port(0x30); //function set 0011 0000 = 8 bits, 1 line, 5x8
    pulse_E();
    DelayMs(5);
    set_Port(0x30);
    pulse_E();
    set_Port(0x30);
    pulse_E();
    set_Port(0x20);
    pulse_E();
    send_command(0x28); //function set 0010 1000 = 4 bits, 2 lines, 5x8
    send_command(0x8); //display on-off 0000 1000 = display-off, cursor-
off, blink-off
    send_command(0x1); // clear display
    send_command(0x6); //Entry mode set 0000 0110 = move right, no
shift
    send_command(0xf); //display on-off 0000 1100 = display-on, cursor-
on, blink-on
    send_command(0x80); //DDRAM en cero (inicio del display) e
incremento por cada caracter escrito
}
  
```

4.2.3.MIWI

Para la conexión con el módulo Wireless se utilizaron varios pines I/O, dado que además de las funciones de conexión inalámbrica, el módulo contiene una EEPROM que viene con la dirección MAC pre-configurada desde la fábrica y espacio disponible para ser usado.



- Bus SPI de comunicación entre MCU, Wireless MIWI y EEPROM interna
 - SCK
 - MOSI
 - MISO
- Physical_Chip_Select: Para comunicación con módulo wireless
- RF_Interrupcion: Salida desde módulo Wireless para solicitar CPU
- Physical_Wake: Para sacar del modo de reposo al módulo Wireless
- RF_EEPROM_Chip_Select: Para comunicación con EEPROM
- Physical_Reset: Para reiniciar módulo wireless

Se necesitan un Chip Select para el módulo Wireless y otro para la EEPROM que también se accede mediante SPI

Figura 4-7: Flujo de inicialización Miwi en End-Device

En caso que el End-Device no logre conexión con la base, éste reiniciará el proceso de inicialización y búsqueda de la red creada por la base.

Una vez inicializado el módulo y establecida la comunicación con la base, se puede iniciar el intercambio de información de acuerdo al programa, que al primer inicio revisa si existen datos grabados en la EEPROM que deban ser enviados a la base.

Posterior a esto, ya sea que existan o no datos almacenados, el End-Device solicita a la base la fecha y hora usando el protocolo propio definido dentro de los mensajes inalámbricos.

La Figura 4-8 muestra el flujo resumido de la aplicación que ejecuta el End-Device.

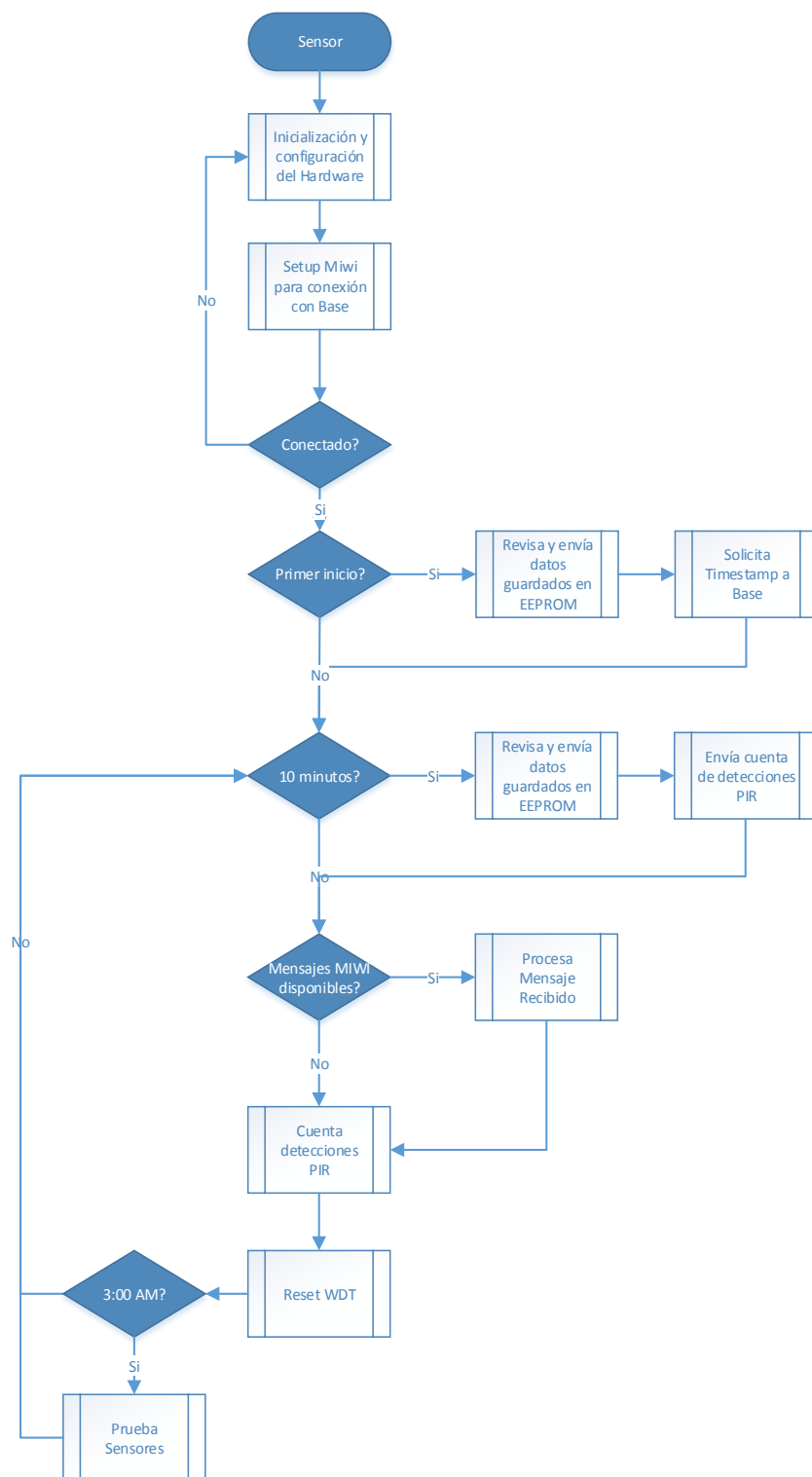


Figura 4-8: Diagrama de flujo aplicación End-Device

Luego de la inicialización y cada 10 minutos el sistema revisa si existen datos pendientes de envío que estén grabados en la EEPROM para poder enviarlos.

Es importante señalar que luego del envío de la información a la base, la página de la EEPROM transmitida es limpiada.

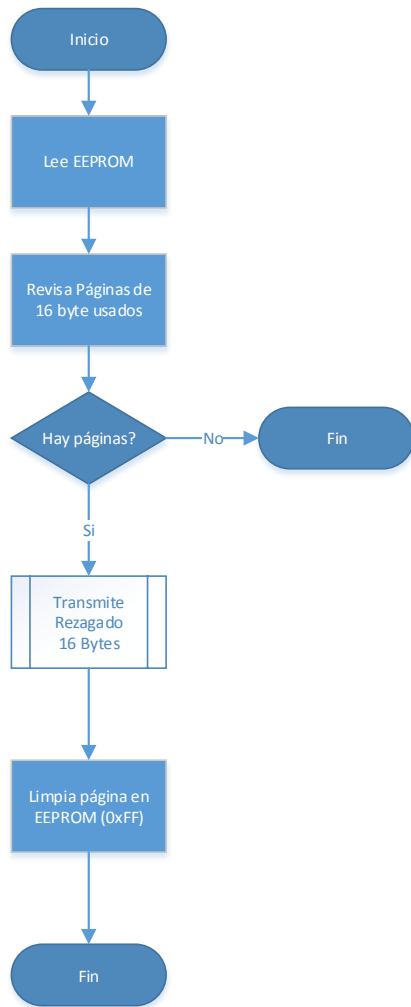


Figura 4-9: Diagrama de flujo control EEPROM

Posteriormente el End-Device envía la cuenta de detecciones del sensor PIR y la medida de luminosidad detectada por el sensor LDR. Para la cuenta de detecciones se verifican todos los cambios de estado que tuvo la entrada asociada, considerando que mientras mayor es la actividad, mayor será el número de disparos del sensor.

El sistema revisa luego si existen mensajes MIWI en memoria, los que al momento de ser recibidos, generan una interrupción que atiende el proceso de recepción y guarda en memoria los datos, junto a actualizar la máquina de estados interna del protocolo.

Los datos recibidos son entonces analizados para tomar alguna acción asociada. El protocolo de mensajes entre la Base y el End-device es el indicado en la Tabla 4-1.

Tabla 4-1: Protocolo de comunicación Base-Device - End-Device

byte	Uso	detalle	Descripcion	Direccion de envio
0	addr [0]		Direccion MAC	Todos
1	addr [1]		Direccion MAC	Todos
2	addr [2]		Direccion MAC	Todos
3	Tipo Mensaje	MENSAJE_DATOS	Mensaje de datos de sensor	enviado desde Sensores a la Base (unicast)
		MENSAJE_TIMESTAMPREQUEST	Solicitud de TimeStamp a la base	enviado desde Sensores a la Base (unicast)
		MENSAJE_TIMESTAMPRESPONSE	Respuesta de TimeStamp	enviado por la Base al sensor (unicast)
		MENSAJE_TIMESTAMPSYNC	Broadcast de TimeStamp para sincronizacion de end-devices	enviado por la Base al sensores (broadcast)
		MENSAJE_REPORTREQUEST	Requerimiento de reporte desde la base a un sensor	enviado por la Base al sensor (unicast)
		MENSAJE_REPORTRESPONSE	Respuesta de reporte	enviado desde Sensores a la Base (unicast)
		MENSAJE_DATOS_PENDIENTES	Envía el contenido de la EEPROM en uno o varios mensajes de 16 bytes	enviado desde Sensores a la Base (unicast)
		MENSAJE_REPORT	Mensaje de reporte de estatus a base	enviado desde Sensores a la Base (unicast)
4	anho			Todos excepto DATOS_PENDIENTES
5	día			Todos excepto DATOS_PENDIENTES
6	mes			Todos excepto DATOS_PENDIENTES
7	hora			Todos excepto DATOS_PENDIENTES
8	díasec			Todos excepto DATOS_PENDIENTES
9	segundo			Todos excepto DATOS_PENDIENTES
10	minuto			Todos excepto DATOS_PENDIENTES
11	Data		Data a ser transmitida, con la lectura de sensores	

MENSAJE_TIMESTAMPRESPONSE: Toma el date time recibido desde la base y lo usa para programar su propio sistema RTCC como se ve en el siguiente segmento de código:

```
void RTCC_Init(void)
{
    INTCONbits.GIEH = 0; //Global Interrupt Enable bit (disable all interrupts)
    EECON2 = 0x55; //secuencia para liberar escritura de registros RTCC
    EECON2 = 0xAA;
    RTCCFGbits.RTCWREN=1;
    RTCCFG = 0xA3;
    RTCVALL = anho; // setea el año
    RTCVALH = 0x0; // Dummy... solo para decrementar RTCCFG.RTCPTR
    RTCVALL = dia; // setea el dia (decena * 16 + unidad)
    RTCVALH = mes; // setea el mes
    RTCVALL = hora; // setea la hora
    RTCVALH = diasem; // setea el dia de la semana (0-6)
    RTCVALL = segundo; // setea los segundos
    RTCVALH = minuto; // setea los minutos
    EECON2 = 0x55; //secuencia para liberar escritura de registros RTCC
    EECON2 = 0xAA;
    RTCCFGbits.RTCWREN=0; // deshabilita la escritura de los registros.
    INTCONbits.GIEH = 1; //Global Interrupt Enable bit (enable all interrupts)
}
```

MENSAJE_TIMESTAMPSYNC: A diferencia del mensaje anterior, este no es solicitado por el End-Device y es enviado como Broadcast desde la base a todos los sensores una vez al día. El procesamiento de la información recibida es idéntico al anterior.

MENSAJE_REPORTREQUEST: Este mensaje es enviado desde la Base para requerir a un End-Device que reporte su estado y la lectura de sus sensores de forma inmediata. Al recibirlo, el End-Device enviara el mensaje ***MENSAJE_REPORTRESPONSE*** con la lectura del sensor PIR, la lectura del sensor LDR y dos Byte con el resultado de la prueba de ambos sensores.

4.2.4. WatchDog Timer

Para la correcta operación del sistema a lo largo del tiempo es fundamental contar con una buena estrategia de recuperación del control del mismo ante eventos de fallas, que pueden llegar a ocurrir por diversos motivos, incluyendo el ruido electromagnético del lugar donde se esté operando. Si bien para sistemas más complejos o críticos, es necesario contar con dispositivos externos y no usar los propios sistemas del MCU, en esta aplicación se usaron los dispositivos propios del MCU.

A fin de prevenir un comportamiento indeseado del sistema se deben tomar una serie de medidas a nivel de hardware externo y de la programación y uso de la MCU

- 1) Usar condensadores de desacople, lo más próximo posible a los pines de alimentación
- 2) Manejo adecuado del MCLR o reset
- 3) Pines I/O no usados deben quedar como salidas, con un valor lógico de cero, conectados a tierra
- 4) Bloqueo de escritura de las áreas de memoria de programa
- 5) Habilitación del WDT por fuses de configuración de la MCU y no por software

La estrategia del WDT debe considerar no solo un paso para reiniciar dicho contador, sino que una secuencia de pasos que el sistema debe cumplir exactamente, o el contador no será reiniciado lo que generará el reset del sistema.

En este diseño se usan dos funciones y una variable global para este control. Al iniciarse el ciclo while del programa principal main() se deja la variable en un valor conocido (WDT_Status=0x55)

```
void wdt_a(void)
{
    if (WDT_Status != 0x55)
    {
        LATBbits.LATB0 = 1;
        while(1);
    }
    WDT_Status += 0x11;
}
```

```
void wdt_b(void)
{
    if (WDT_Status != 0x88)
    {
        LATBbits.LATB0 = 1;
        while(1);
    }
    if (WDT_Status != 0x88)
    {
        LATBbits.LATB0 = 1;
        while(1);
    }
    ClrWdt();
    WDT_Status = 0x00;
}
```

Inmediatamente luego de setear la variable en el valor 0x55, se ejecuta la función `wdt_a()`;

Al final del ciclo `while`, cuando el sistema ya ejecuto todas las acciones programadas se incrementa la variable (`WDT_Status+=0x22`) y se ejecuta la función `wdt_b()`, la que pregunta dos veces por el valor de la variable antes de reiniciar el contador del watchdog timer. De esta forma nos aseguramos que ante eventos como el escape del programa, existan muy pocas posibilidades de que esto no afecte la estrategia de detección y control implementada. Nótese que si alguno de los métodos `wdt` falla en la detección correcta del valor de la variable, el programa quedará detenido esperando el reset del WDT y encenderá un led que permita ver que esto ocurrió en el momento.

4.2.5.Timer0

Para el control de los tiempos dentro del sistema se utilizó el timer0 mediante la función **MiWi_TickGet(void)**, la que retorna la estructura de tiempo.

```
MIWI_TICK MiWi_TickGet(void)
{
    MIWI_TICK currentTime;
    BYTE IntFlag1;
    BYTE IntFlag2;

    /* copy the byte extension */
    currentTime.byte.b2 = 0;
    currentTime.byte.b3 = 0;

    /* disable the timer to prevent roll over of the lower 16 bits while before/after reading of the extension */
    TMR_IE = 0;

    do
    {
        IntFlag1 = TMR_IF;
        vTickReading[0] = TMR0L;
        vTickReading[1] = TMR0H;
        *((DWORD*)&vTickReading[2]) = dwInternalTicks;
        IntFlag2 = TMR_IF;
    } while(IntFlag1 != IntFlag2);
    currentTime.byte.b0 = vTickReading[0];
    currentTime.byte.b1 = vTickReading[1];

    if( IntFlag1 > 0 )
    {
        TMR_IF = 0;
        dwInternalTicks++; // DQ para ENC que en lugar de 4 bytes para Ticks usa 6 bytes
        timerExtension1++;
        if(timerExtension1 == 0)
        {
            timerExtension2++;
        }
    }

    /* copy the byte extension */
    currentTime.byte.b2 += timerExtension1;
    currentTime.byte.b3 += timerExtension2;

    /* enable the timer*/
    TMR_IE = 1;

    return currentTime;
}
```

La configuración del timer se aprecia en la función de inicialización **InitSymbolTimer()**

```
void InitSymbolTimer()
{
    TMR_CON = 0b00000000 | CLOCK_DIVIDER_SETTING;
    TMR_IP = 1;
    TMR_IF = 0;
    TMR_IE = 1;
    TMR_ON = 1;

    timerExtension1 = 0;
    timerExtension2 = 0;
    dwInternalTicks = 0; // ENC
}
```

En esta se ve que el registro T0CON (TMR_CON) es cargado con el divisor adecuado de 1:32 para operar en 8Mhz que es la frecuencia usada para el sistema, dejándolo en modo 16 bits, usando el reloj interno como fuente de su frecuencia (modo timer)

```
#elif(CLOCK_FREQ <= 8000000)
#define CLOCK_DIVIDER 32
#define CLOCK_DIVIDER_SETTING 0x04
#define SYMBOL_TO_TICK_RATE 8000000
```

Una vez recibido el valor retornado, este es restado del valor anterior recibido con el que se desea comparar la lectura, y este resultado es comprado contra constantes de tiempo definidas.

```
#define ONE_SECOND (((DWORD)CLOCK_FREQ/1000 * 62500) / (SYMBOL_TO_TICK_RATE / 1000))
#define ONE_MILI_SECOND (ONE_SECOND/1000)
#define HUNDRED_MILI_SECOND (ONE_SECOND/10)
#define FORTY_MILI_SECOND (ONE_SECOND/25)
#define TWENTY_MILI_SECOND (ONE_SECOND/50)
#define TEN_MILI_SECOND (ONE_SECOND/100)
#define FIVE_MILI_SECOND (ONE_SECOND/200)
#define TWO_MILI_SECOND (ONE_SECOND/500)
#define ONE_MINUTE (ONE_SECOND*60)
#define ONE_HOUR (ONE_MINUTE*60)
```

El valor ONE_SECOND es la definición de referencia que se usa en el resto de las definiciones y esta usa la frecuencia del clock del sistema, y las definiciones asociadas a la frecuencia del procesador. Para el caso de este sistema se tiene:

$$ONE_SECOND = \frac{\left(\left(\frac{8.000.000}{1000}\right) * 62500\right)}{(8.000.000 / 1000)} \text{ [Ciclos de instrucción a 8MHz]}$$

Es decir que ONE_SECOND corresponde a 62500 ciclos de instrucción a 8 MHz con divisor de 32. Considerando que el ciclo de instrucción toma 4 clock, tenemos:

$$ONE_{SECOND} = \frac{8.000.000}{32 * 4} = 62500 \text{ ciclos de instrucción}$$

Para mayor claridad, el diagrama de bloques interno del Timer 0 se muestra en la Figura 4-10

FIGURE 12-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)

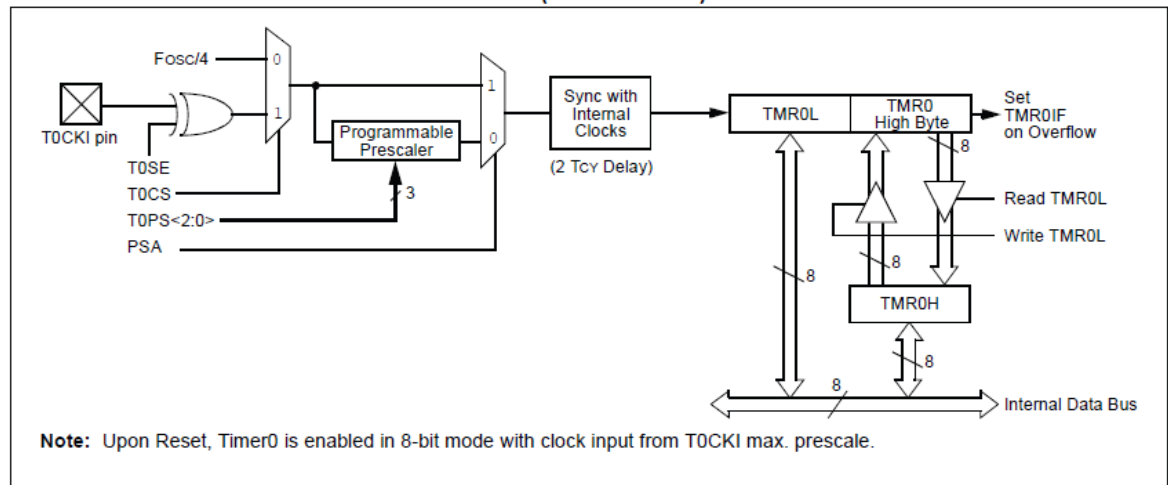


Figura 4-10: Diagrama de bloques Timer 0

4.2.6.Linker

Debido a la necesidad de generar estructuras de datos mayores a 256 bytes y a la limitación nativa de la MCU de sus bancos de memoria se hace necesario modificar el archivo nativo del linker de Microchip para generar un banco de memoria “virtual” que traspase este límite físico.

El archivo que debió ser editado es el correspondiente al procesador, en este caso el 18f27j53_g.lkr

Original				Modificado			
DATABANK	NAME=gpr0	START=0x60	END=0xFF	DATABANK	NAME=gpr0	START=0x60	END=0xFF
DATABANK	NAME=gpr1	START=0x100	END=0x1FF	DATABANK	NAME=gpr1	START=0x100	END=0x1FF
DATABANK	NAME=gpr2	START=0x200	END=0x2FF	DATABANK	NAME=gpr2	START=0x200	END=0x2FF
DATABANK	NAME=gpr3	START=0x300	END=0x3FF	DATABANK	NAME=gpr3	START=0x300	END=0x3FF
DATABANK	NAME=gpr4	START=0x400	END=0x4FF	DATABANK	NAME=gpr4	START=0x400	END=0x4FF
DATABANK	NAME=gpr5	START=0x500	END=0x5FF	DATABANK	NAME=gpr5	START=0x500	END=0x5FF
DATABANK	NAME=gpr6	START=0x600	END=0x6FF	DATABANK	NAME=gpr6	START=0x600	END=0x6FF
DATABANK	NAME=gpr7	START=0x700	END=0x7FF	DATABANK	NAME=gpr7	START=0x700	END=0x7FF
DATABANK	NAME=gpr8	START=0x800	END=0x8FF	DATABANK	NAME=gpr8	START=0x800	END=0x8FF
DATABANK	NAME=gpr9	START=0x900	END=0x9FF			PROTECTED	
DATABANK	NAME=gpr10	START=0xA00	END=0xAFF	DATABANK	NAME=gpr9	START=0x900	END=0x9FF
DATABANK	NAME=gpr11	START=0xB00	END=0xBFF			PROTECTED	
				DATABANK	NAME=gpr10b	START=0xA00	END=0xB7F
						PROTECTED	
				DATABANK	NAME=gpr11	START=0xB80	END=0xBFF
				SECTION NAME=TCB_uRAM		RAM=gpr10b	
				SECTION NAME=TempV		RAM=gpr8	
				SECTION NAME=MyMem		RAM=gpr9	

La modificación consiste en indicarle al Linker la creación de una sección de memoria protegida o PROTECTED para que el procesador no la considere para el uso de variables comunes (gpr8 y gpr9) y además la modificación de los límites de un banco de memoria, como en el caso de gpr10b, donde se aprecia que gpr10b aumentó su tamaño desde 256 bytes hasta 383 bytes, y el gpr11 que disminuyó a 128 bytes

Luego en la aplicación fueron usados los nombres definidos en los “SECTION NAME” unidos a un pragma del compilador de forma de indicar al compilador-linker que use estos bancos específicos para las variables en cuestión

```
#pragma udata TCB_uRAM
static TCB_STUB TCBStubs[TCP_SOCKET_COUNT]
#pragma udata

#pragma udata MyMem
BYTE MyDataMEM_RX[256];
#pragma udata

#pragma udata TempV
struct TempValues NodeTemp[MAX_NODES];
#pragma udata
```

4.2.7.Sensores

Para los sensores se ha elegido un detector PIR de una salida on-off que puede dispararse en modo continuo o simple mediante un jumper de ajuste, con un rango de alimentación de 3.3v a 5.0v. Para nuestra aplicación usaremos el modo simple (pasa de low a high y se mantienen en esta condición mientras esté disparado) y contaremos en el MCU la cantidad de cambios que se reciban en la entrada asociada.

El rango de detección es de hasta 10 metros, lo que lo hace ideal para ubicarlo en el interior de la cabaña, en un área de uso común como el comedor, y apuntándolo hacia la dependencias interiores.

Adicionalmente se ha usado un detector de luminosidad LDR que está conectado a una entrada configurada como conversor A/D mediante un divisor de tensión, que en definitiva nos indicara el nivel de luminosidad del entorno.

La idea es contrastar estas mediciones en el webserver que reciba esta información de forma que pueda detectar patrones como:

- Hora del día v/s actividad detectada en PIR
 - Indicador de uso de la unidad
 - Indicador de actividad de aseo (dependiendo de la hora programada y duración)
- la luminosidad recibida en relación a la hora del día
 - Indicador de que el End-Device puede estar tapado
 - Indicador de uso de la unidad

4.2.8. Sensores alternativos

Además de estos sensores se pueden agregar detectores de consumo de energía eléctrica, detectores de flujo para detectar consumo de agua, detectores de temperatura y humo para alertar rápidamente en caso de incendio, etc. Las posibilidades dependen de la cantidad de variables que se deseen controlar. Para el proyecto aquí indicado, consideramos muy sensible el tema del precio, por lo que es prioridad buscar el menor costo posible de setup, por lo que no se incluyó el uso de estos sensores, no obstante se podrían ofrecer como un adicional.



Figura 4-11: Sensores alternativos

4.3. Implementación de la Base

Los módulos descritos en el desarrollo del End-Device son los mismos usados en el Base-Device por lo que no serán repetidos, no obstante el flujo del programa es diferente y existe el módulo adicional de conexión Ethernet.

En la Figura 4-12 se aprecia el prototipo del Base-Device construido.

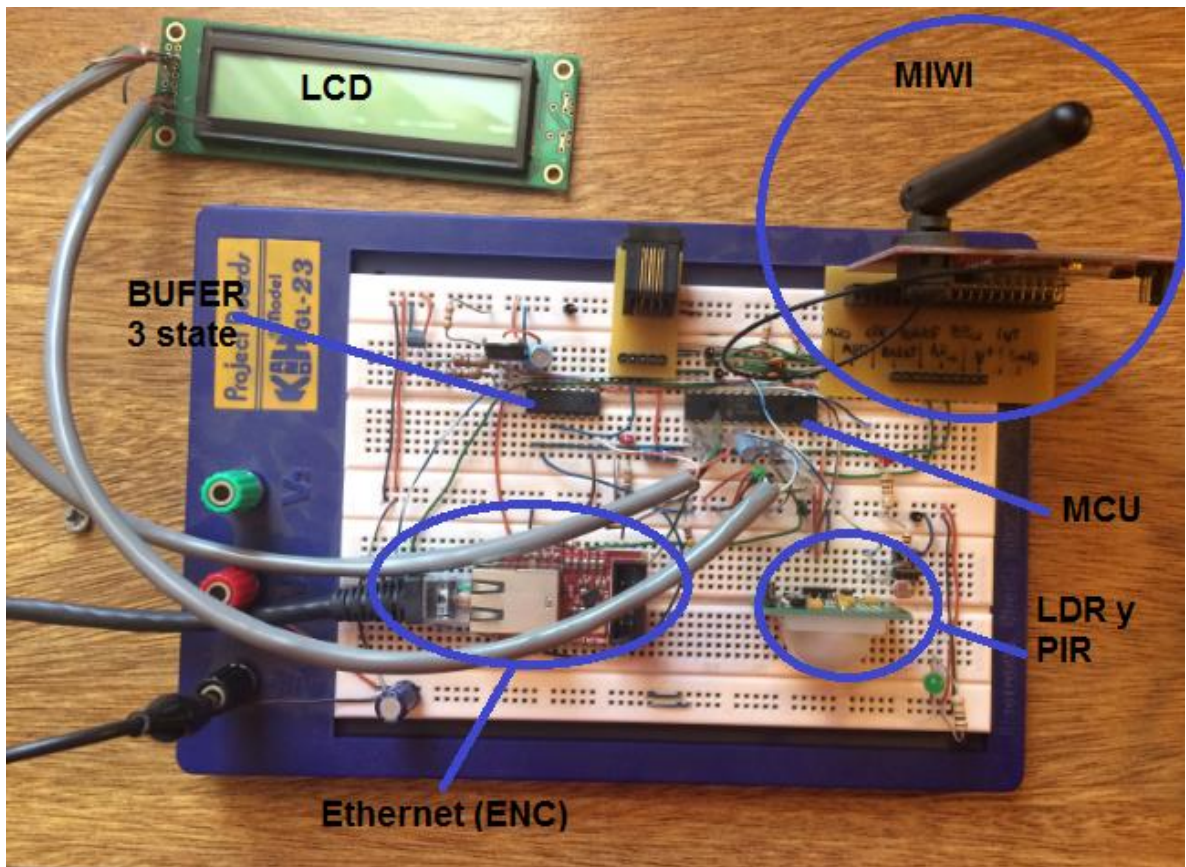


Figura 4-12: Base-Device

4.3.1.MCU

Además de las tareas ya descritas en el End-Device, la base debe deshabilitar la interrupción INT1 y usar pooling en lugar de interrupciones para revisar el estado de los periféricos y se debe configurar el módulo ENC e inicializar los protocolos de red.

La Figura 4-13 muestra las acciones de la inicialización del sistema

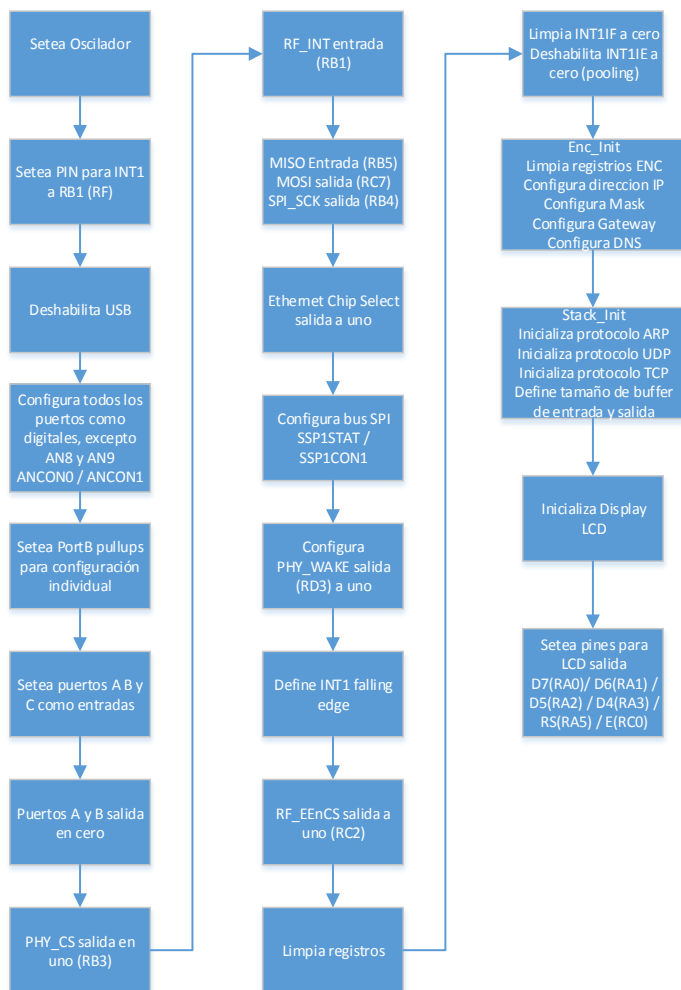


Figura 4-13: Flujo de inicialización Base-Device

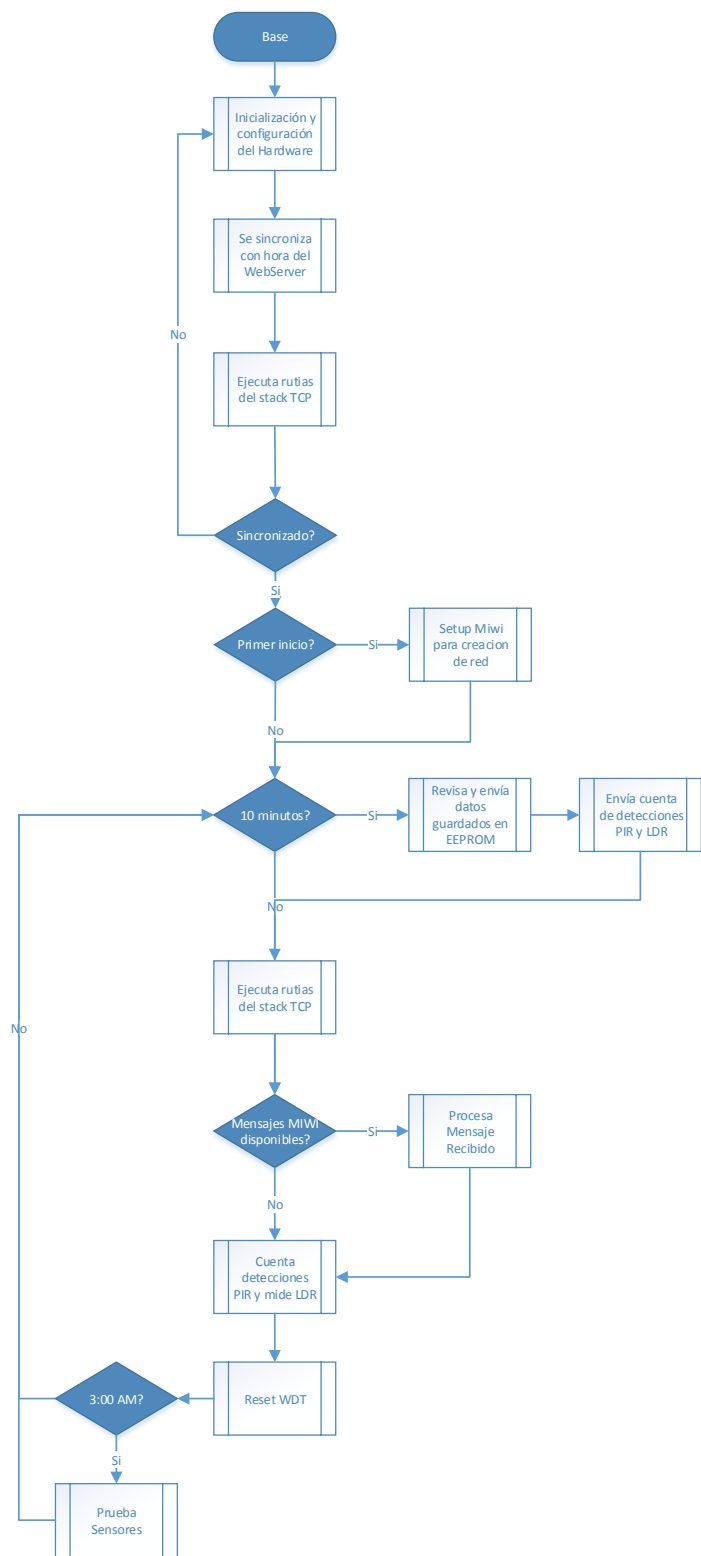


Figura 4-14: Diagrama de flujo aplicación Base-Device

Luego de la inicialización del hardware, el Base-Device intenta obtener la fecha y hora desde el WebServer, para lo cual ejecuta el PHP adecuado que le devuelve estos valores. En caso de no obtenerlos se mantiene en loop intentándolo (ver Figura 4-14)

Durante este proceso de envío y recepción de mensajes TCP es necesario ejecutar lo más frecuentemente posible las funciones que actualizan la máquina de estados del stack tcp, pues durante el proceso se pasa por varios estados y se envían varios mensajes entre las partes que se están comunicando.

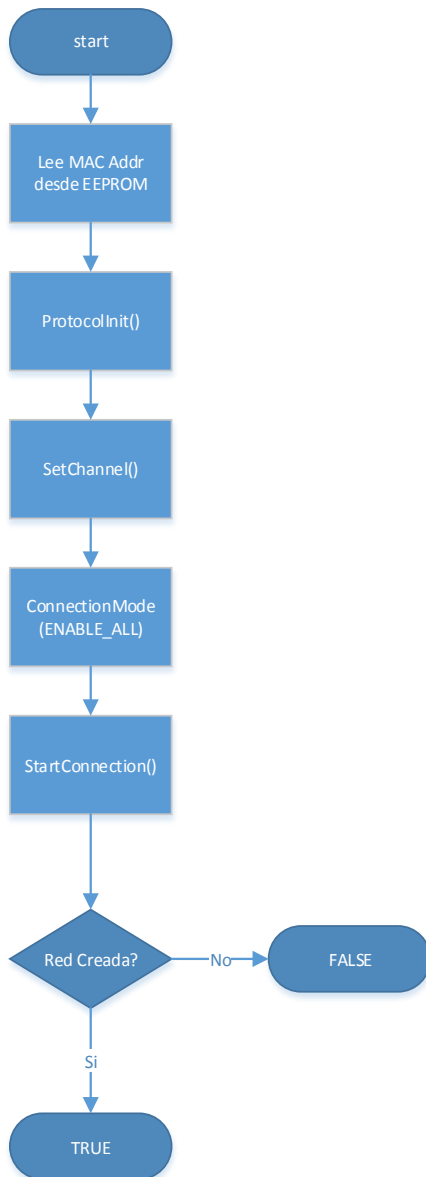
Luego de sincronizarse con el WebServer, el Base-Device crea la red Miwi para recibir conexiones desde los End-Devices.

En ciclos de 10 minutos hace la revisión de datos guardados en EEPROM y la de sus propios sensores y de todas las mediciones que los End-Device le han transmitido y los envía por TCP hacia el WebServer, agregando su propia Mac address y la de los End-Devices para que posteriormente sean identificados unívocamente.

La prueba de sus sensores ocurre al igual que en el End-Device a las 3:00 am y la rutina de WDT está presente permanentemente en el ciclo normal de ejecución.

4.3.2.MIWI

Para la conexión con el módulo Wireless se utilizaron varios pines I/O, dado que además de las funciones de conexión inalámbrica el módulo contiene una EEPROM que viene con la dirección MAC configurada desde la fábrica y espacio disponible para ser usado.



En el Base-device se debe crear la red MIWI, y esperar la conexión desde los End-devices. Se selecciona uno de los canales que esta previamente acordado y se inicia la escucha de tráfico en el mismo.

El Base-device a diferencia del End-Device no usa el método de interrupciones para atender los requerimientos de este módulo y en lugar de esto se usa el método de pooling de la variable que señala la interrupción INT1IF. Esto debido a que el bus SPI es compartido con el módulo ENC de comunicación Ethernet, y está por tanto en permanente uso.

Una interrupción del módulo MIWI durante el intercambio de información entre la MCU y el módulo ENC se traduce en el truncamiento de este intercambio y la pérdida de la máquina de estados de los protocolos de red.

Figura 4-15: Flujo de inicialización Miwi en Base-device

4.4. Desarrollo del WebServer de Prueba

Para poder probar el funcionamiento del sistema fue necesaria la creación de un sitio web muy simple que expusiera paginas php para uso del sistema y además de una base de datos MySQL que mediante estas interfaces pudiera registrar los datos aportados por el sistema probado.

Las páginas usadas por el Base-Device son las siguientes:

Procesa5.php

La función de esta página es entregar la fecha y la hora a las bases que se conecten.

Procesa3.php

La función de esta página es recibir los datos desde el Base-Device y registrarlos en la base de datos MySQL

P6.php

La función de esta página es recibir los datos rezagados desde el Base-Device y registrarlos en la base de datos MySQL

4.5. Prueba del sistema y Mediciones de campo

Para las mediciones de campo se configura un End-Device y un Base-Device. Este último será conectado mediante un cable Ethernet cat5e directamente a un notebook donde se encuentra el sitio web desarrollado en PHP para recibir los datos desde el Base-Device.

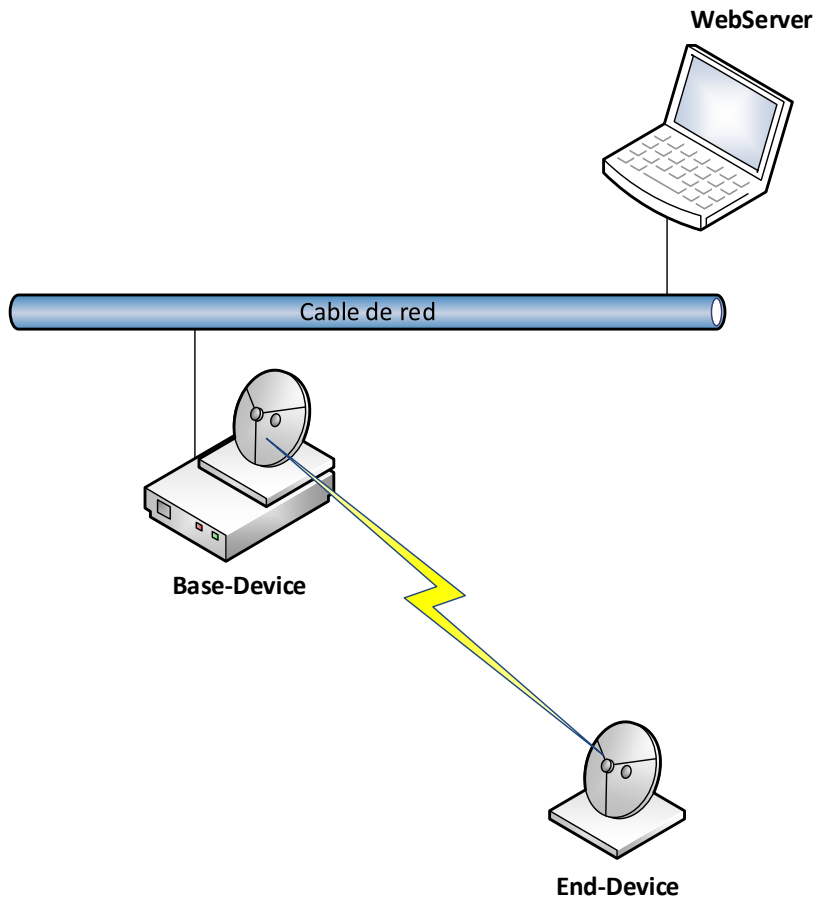


Figura 4-16: Diagrama del ambiente de testing

Las mediciones obtenidas en campo se realizaron estableciendo el link de comunicación del End-Device con el Base-Device y luego alejarse portando el End-Device hasta que se perdiera el link de comunicación, lo que es señalado en el LCD del equipo. Estas mediciones indican un alcance de hasta 120 metros en línea de vista y de 80 metros con obstrucción completa de vista, lo que cumple con los requerimientos de alcance necesarios para el proyecto.

Durante las pruebas de desarrollo y con el fin de tener un feedback más rápido de la actividad del sistema se cambia la configuración para que las mediciones ocurran cada 10 segundos en lugar de 10 minutos. Una vez que se logran probar todas las funcionalidades

requeridas en esta modalidad se le deja funcionando durante un día con el ciclo normal de 10 minutos. Los resultados obtenidos son los mostrados en la Figura 4-17

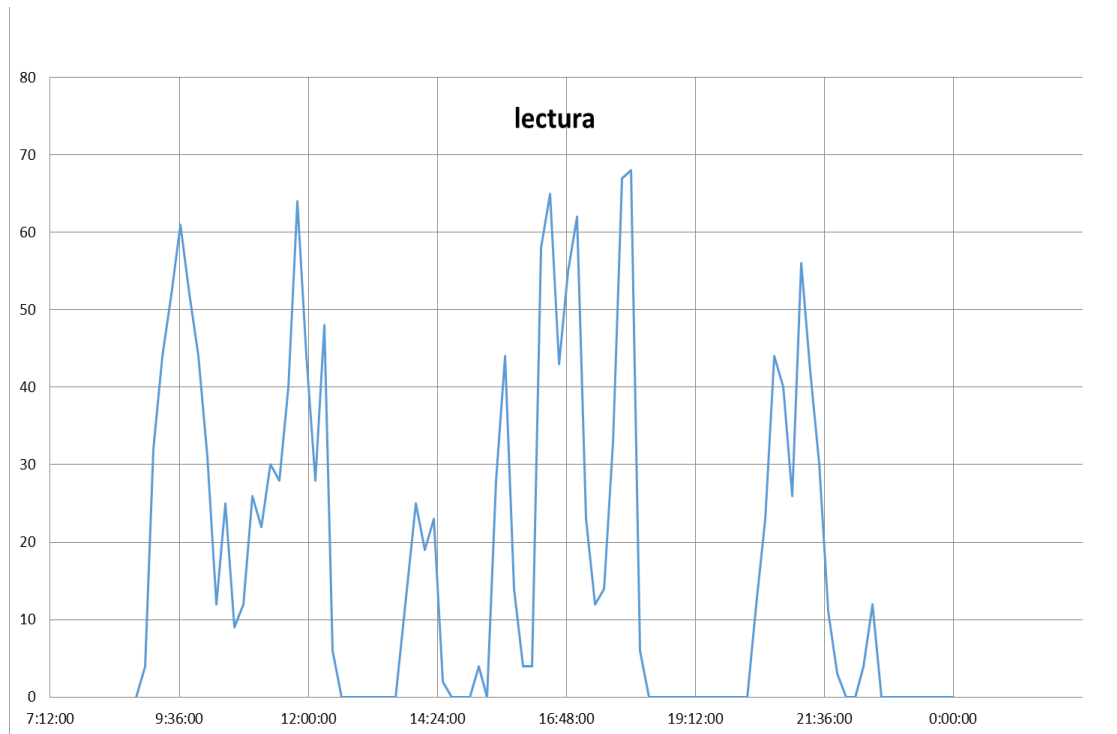


Figura 4-17: Gráfico de mediciones de ocupación obtenidas del sensor PIR

Para este caso el End-Device fue puesto en el comedor un día domingo, reflejando la actividad familiar del día. En este se ve actividad entre las 9:00 hrs y las 22:30 hrs, dado que fuera de este horario no hubo nadie en el comedor. Dentro del día se aprecia falta de actividad ente las 12:30 y 13:30 hrs y entre las 18 hrs y 20 hrs dado que la casa no estaba con ocupantes.

Adicionalmente se muestra en la Figura 4-18 la medición del sensor LDR durante el mismo lapso de tiempo, el que nos muestra la curva de luz natural de la zona donde estaba ubicado el sensor sumado al efecto de la luz artificial usado entre las 20:00 hrs y 23:00 hrs aproximadamente. Observando esta imagen resulta natural suponer la ocupación de una cabaña por el simple hecho de detectar la presencia de luz artificial en horario nocturno, no obstante esto no es suficiente pues podría tratarse de una unidad vacía pero con una luz encendida.

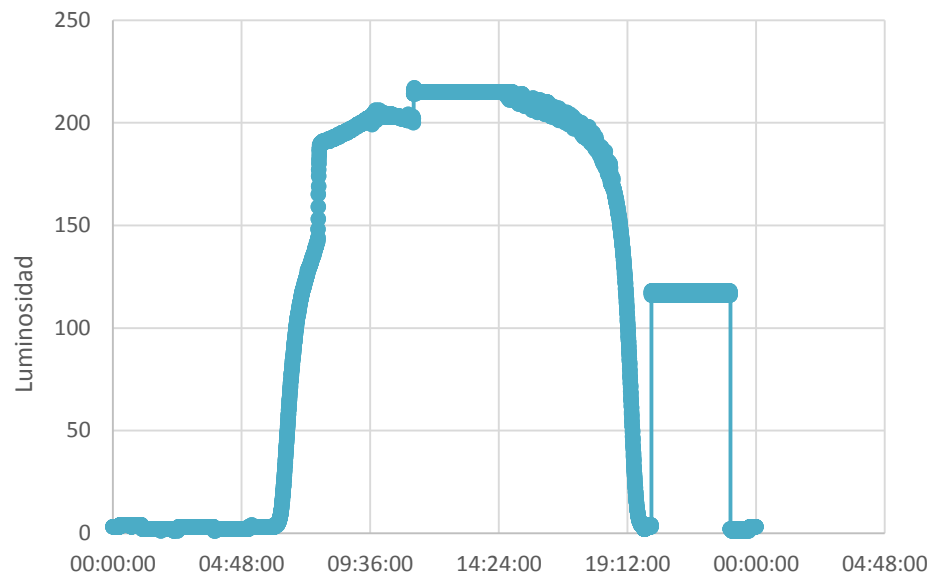


Figura 4-18: Gráfico de mediciones obtenidas del sensor LDR

Finalmente uniendo ambas imágenes en la Figura 4-19, se puede apreciar la correlación entre las detecciones del sensor PIR y LDR, donde en el horario de noche, ambos sensores muestran un “disparo” de sus lecturas, y durante el día, es el sensor PIR quien descubre los movimientos efectuados dentro del recinto.

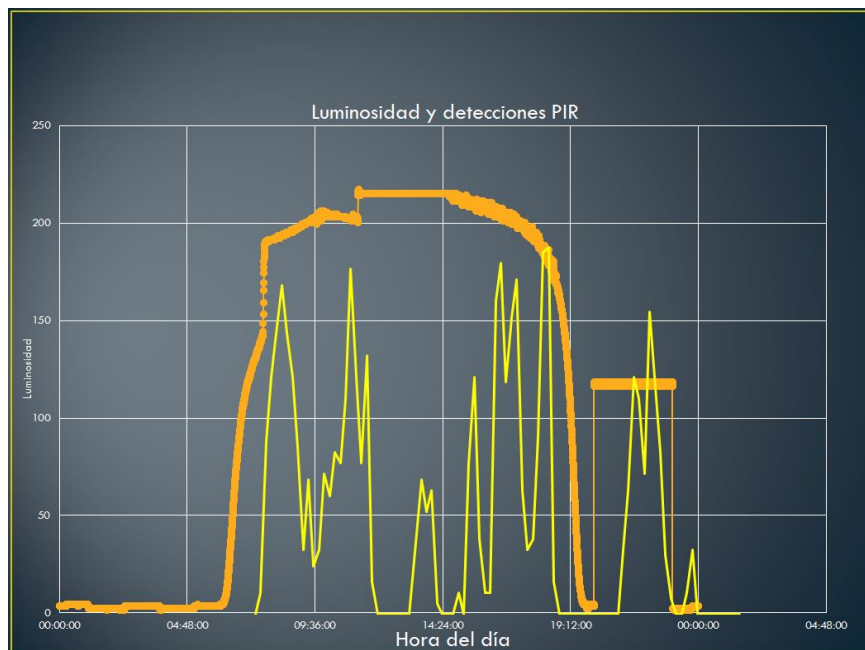


Figura 4-19: Correlación detección PIR y LDR

Mediante capturas de tráfico usando wireshark se observa que el sistema envía correctamente la información al servicio web, lo que se puede apreciar en la Figura 4-20

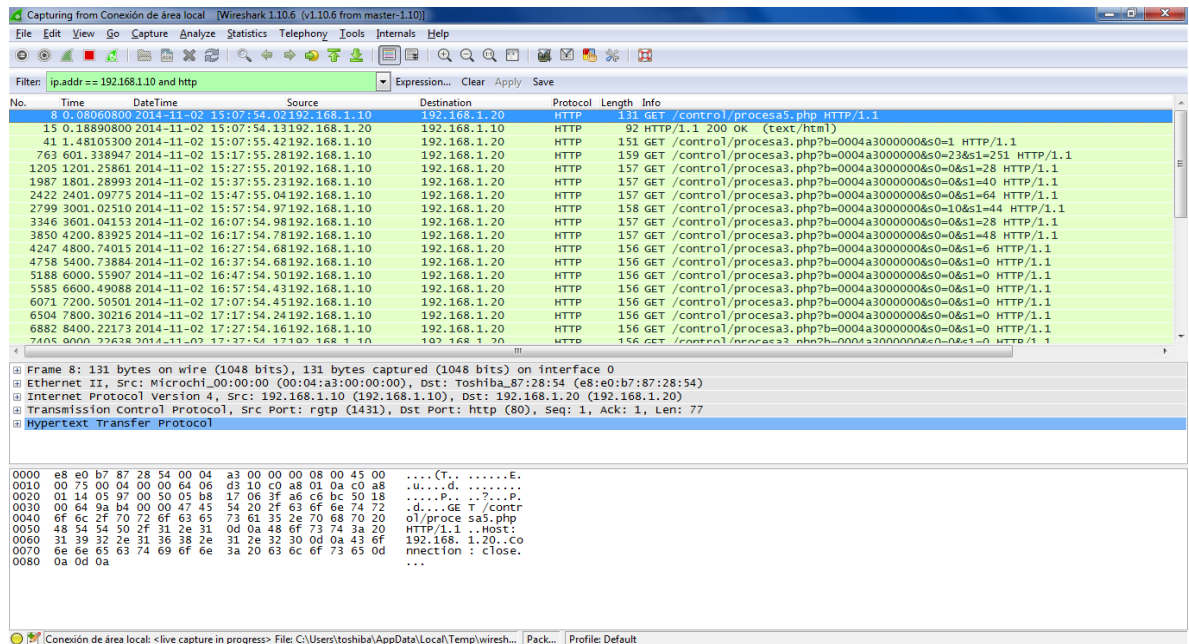


Figura 4-20: Captura de tráfico de red entre Base-Device y Web Server

Usando el osciloscopio se verifica el funcionamiento de la comunicación del MCU con los módulos Ethernet y Miwi mediante el bus SPI.

En la Figura 4-21 se muestra el canal1 MISO y canal2 SCK, con la frecuencia programada de 500khz

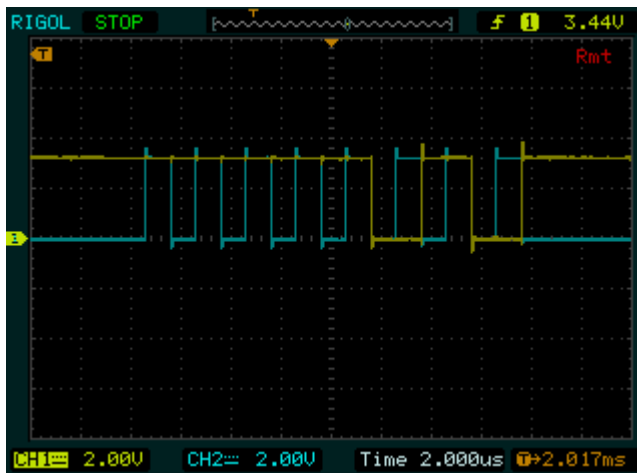


Figura 4-21: Señales MISO (en amarillo) y SCK (en azul) del bus SPI

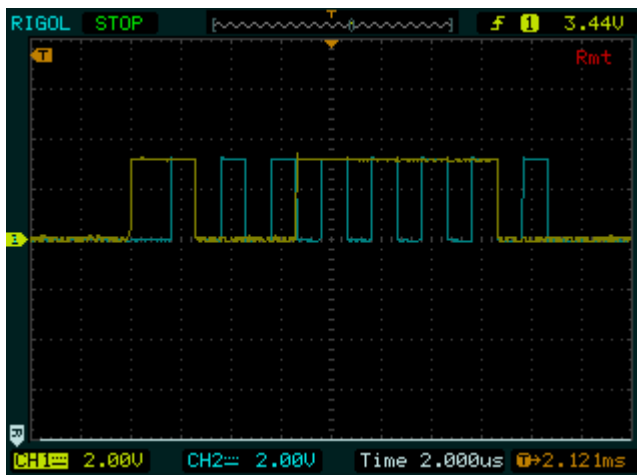


Figura 4-22: Señales MOSI y SCK del bus SPI

En la Figura 4-22 se muestra el canal1 MOSI en amarillo y canal2 SCK en azul, con la frecuencia programada de 500khz

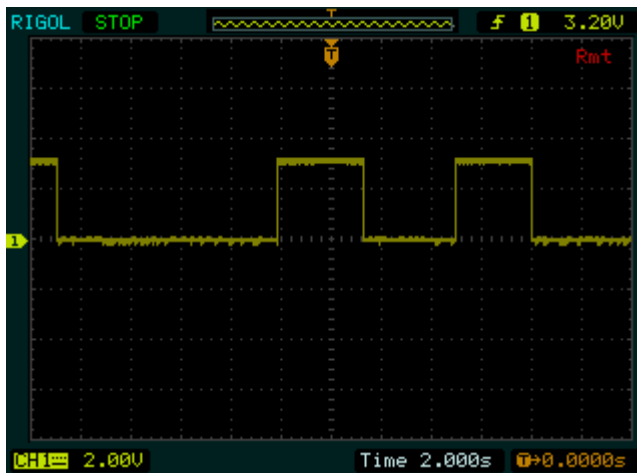


Figura 4-23: Señal de salida del módulo PIR

El sensor PIR funciona dejando su salida a nivel alto cuando ocurre una detección por un período de cerca de 2,5 a 3 segundos luego de lo cual regresa al nivel bajo. Si se produce un movimiento permanente, el sensor permanece en nivel alto y luego de finalizada la detección, permanece en ese estado por cerca de 1 segundo para regresar al nivel bajo, lo que se muestra en la Figura 4-23

Por lo anterior, y dado que el funcionamiento normal del sensor es generar más disparos mientras más actividad hay, la cuenta de dichos cambios genera una medida razonablemente alineada con el nivel de ocupación o tránsito generado en la zona de detección.

5. Conclusiones y Trabajo Futuro

Con la experiencia del diseño y desarrollo del proyecto descrito podemos concluir que es posible lograr el objetivo de obtener la medición de la ocupación de cabañas mediante la combinación de un servicio web de procesamiento y despliegue de información y de un sistema de hardware que provea la medición de presencia automática en terreno mediante diversos tipos de sensores.

Todos los requisitos inicialmente planteados fueron logrados mostrándose muy útil el sensor de movimiento para obtener una medición real de la ocupación de las cabañas. Otro punto importante es que debido a la característica de red inalámbrica el montaje del sistema es fácil y no invasivo, lo que era parte de los requisitos iniciales del proyecto. De esta manera se demuestra la utilidad de los protocolos basados en IEEE 802.15.4 para esta aplicación.

Un punto que se debe mejorar es el módulo de comunicación Ethernet, ya que si bien el ENC28j60 escogido es de fácil utilización, solo funciona en velocidad 10mbps sin contar con auto negociación del dúplex, lo que significa que el switch o equipo donde se conecte debe ser manualmente seteado.

Para lograr pasar desde la etapa de prototipo a un modelo de producción es necesario mejorar varios aspectos incluido el costo de cada unidad, además de lograr el desarrollo de un sistema de reserva y pago de cabañas integrado al control de ocupación, de modo de poder ofrecer el servicio completo bajo un modelo de negocio alineado con las necesidades de los dueños, ya sea como un costo adicional por el arriendo o un fijo mensual por cada unidad.

6. Referencias

- [1] Ibrahim Dogan, Advanced PIC Microcontroller Projects in C. 1a edición, USA, Newnes, 2008. 560p
- [2] Bates Martin, Interfacing Pic Microcontrollers 1a edición, USA, Newnes, 2006. 313p
- [3] Bates Martin, Programming 8-bit PIC Microcontrollers in C 2008 1a edición, USA, Newnes, 2008. 278p
- [4] Edwards Lewin, So You Wanna Be an Embedded Engineer 1a edición, USA, Newnes, 2006. 256p
- [5] Smith David, PIC in Practice 3rd Edition, USA, Newnes, 2013. 352p
- [6] Wilmshurst Tim, Designing Embedded Systems with PIC Microcontrollers 2nd Edition, USA, Newnes, 2009. 750p
- [7] Vishram S. Pandit; Woong Hwan Ryu; Myoung Joon Choi, Power Integrity for I/O Interfaces: With Signal Integrity/Power Integrity Co-Design 1a edición, USA, Prentice Hall, 2010. 416p
- [8] Darren Ashby; Bonnie Baker; Ian Hickman; Walt Kester; Robert Pease; Tim Williams; Bob Zeidman, Circuit design: Know it all, 2dn edition, Newnes 2011. 1248p
- [9] John Catsoulis, Designing Embedded Hardware, 2nd Edition, USA, O'Reilly 2005. 398p
- [10] Microchip. PIC18F47J53 Family Data Sheet, USA, Microchip, 2010. 586p
- [11] Microchip. MRF24J40 Data Sheet, USA, Microchip, 2010. 586p
- [12] Yifeng Yang, Wireless (MiWi™) Media Access Controller – MiMAC Mirochip, USA, Microchip 2009. 24p
- [13] Yifeng Yang, Microchip Wireless (MiWi™) Application Programming Interface – MiApp, USA, Microchip 2009. 24p
- [14] Yifeng Yang, Microchip MiWi™ PRO Wireless Networking Protocol, USA, Microchip 2009. 30p
- [15] Microchip. ENC28j60 Stand-Alone Ethernet Controller with SPI Interface, USA, Microchip 2012. 102p
- [16] Nilesh Rajbharti, The Microchip TCP/IP Stack, USA, Microchip 2008. 94p
- [17] Microchip, Microchip TCP/IP Stack Help, USA, Microchip 2011. 648p
- [18] M. Simmons, Ethernet Theory of Operation, USA, Microchip 2008. 26p
- [19] Olimex. ENC28J60-H development board Users Manual, Mexico, Olimex 2012. 11p