



ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
ESCUELA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y REDES
INDUSTRIALES

“DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE
MONITOREO DE FLUJO DE AGUA MEDIANTE GSM Y RADIO
FRECUENCIA EN UN CIRCUITO LOCAL DE AGUA POTABLE”

TRABAJO DE TITULACIÓN: **DISPOSITIVOS TECNOLÓGICOS**
para optar al Grado Académico de:
INGENIERO EN ELECTRÓNICA, CONTROL Y REDES
INDUSTRIALES

AUTOR:
WILMER VINICIO GUAÑO OCHOA

Riobamba – Ecuador
2017

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
ESCUELA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y REDES
INDUSTRIALES

El tribunal del trabajo de titulación certifica que el: **“DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO DE MONITOREO DE FLUJO DE AGUA MEDIANTE GSM Y RADIO FRECUENCIA EN UN CIRCUITO LOCAL DE AGUA POTABLE”**, de responsabilidad del señor WILMER VINICIO GUAÑO OCHOA, ha sido minuciosamente revisado por los miembros del tribunal del trabajo de titulación, quedando autorizada su presentación.

NOMBRE	FIRMA	FECHA
Ing. Washington Luna. E DECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRÓNICA.
Ing. Freddy Chávez. V DIRECTOR DE LA ESCUELA DE INGENIERÍA ELECTRÓNICA EN CONTROL Y REDES INDUSTRIALES.
Ing. Edwin Altamirano. S DIRECTOR DE TRABAJO DE TITULACIÓN
Ing. Wilson Zúñiga. V MIEMBRO DEL TRIBUNAL.

TABLA DE CONTENIDO

ÍNDICE DE TABLAS.....	vi
ÍNDICE DE FIGURAS.....	viii
ÍNDICE DE GRÁFICOS.....	xi
RESUMEN.....	xiv
ABSTRACT.....	xv
INTRODUCCIÓN	1
CAPITULO I	
1. MARCO TEÓRICO REFERENCIAL	5
1.1 Agua.....	5
1.2 Monitorización	6
1.2.1 <i>Proceso de monitorización</i>	6
1.2.2 <i>Monitorización del agua potable</i>	7
1.3 Redes de distribución de agua.....	7
1.3.1 <i>Estructura jerárquica</i>	7
1.3.2 <i>Tipologías</i>	8
1.3.2.1 <i>Red ramificada o circuito abierto</i>	8
1.3.2.2 <i>Red mallada o circuito cerrado</i>	9
1.3.2.3 <i>Características de las redes de distribución de agua potable</i>	9
1.4 Red de sensores inalámbricos (WSN).....	10
1.4.1 <i>Hardware de un nodo sensor</i>	11
1.4.2 <i>Tipologías con la red con sensores</i>	12
1.4.3 <i>Áreas de aplicación</i>	13
1.5 Comunicación inalámbrica	15
1.5.1 <i>Clasificación por rangos de frecuencia</i>	15
1.5.1.1 <i>Infrarrojos (IR)</i>	15
1.5.1.2 <i>Microondas (MW)</i>	15
1.5.1.3 <i>Radiofrecuencias (RF)</i>	16
1.6 Tarjetas de desarrollo.....	16
1.6.1 <i>Arduino</i>	16

1.6.1.1	<i>Shield</i>	17
1.7	Raspberry Pi	18
1.7.1	<i>Galileo</i>	19
1.8	Sensores	21
1.9	Desarrollo de páginas web	22
1.9.1	<i>Lenguajes de programación</i>	22
1.9.2	<i>Estructura del sitio web</i>	24
1.9.2.1	<i>Encabezado</i>	24
1.9.2.2	<i>Barra de navegación</i>	24
1.9.2.3	<i>Área de contenido principal</i>	24
1.9.2.4	<i>Barra lateral de contenido</i>	25
1.9.2.5	<i>Pie de página</i>	25
1.10	Base de datos	25
1.10.1	<i>¿Por qué se utiliza una base de datos?</i>	26

CAPITULO II

2.	MARCO METODOLÓGICO	27
2.1	Introducción	27
2.2	Requerimientos del prototipo	27
2.3	Requerimientos de la arquitectura del prototipo	28
2.3.1	<i>Módulo de Lectura, Transmisión y Activación</i>	29
2.3.2	<i>Módulo de Recolección, Visualización y Control</i>	29
2.4	Diseño de módulos del prototipo	30
2.4.1	<i>Módulo de Lectura, Transmisión y Activación</i>	30
2.4.2	<i>Módulo de Recolección, Visualización y Control</i>	31
2.5	Selección de dispositivos para el desarrollo del prototipo	32
2.5.1	<i>Tarjetas de desarrollo</i>	33
2.5.2	<i>Comunicación</i>	35
2.5.2.1	<i>Módulo de comunicación nRF24L01</i>	35
2.5.2.2	<i>Módulo de comunicación Sim800L</i>	36
2.5.2.3	<i>Módulo de comunicación ethernet shield</i>	37
2.5.3	<i>Sensor de flujo para el prototipo</i>	38
2.5.4	<i>Actuadores de control para el prototipo</i>	40
2.5.5	<i>Alimentación del prototipo</i>	41
2.6	Conexiones del prototipo	43
2.6.1	<i>Esquema de conexión del módulo Lectura, Transmisión y Activación</i>	44
2.6.2	<i>Esquema de conexión del módulo Recolección, Visualización y Control</i>	46

2.7	Desarrollo del código del prototipo	48
2.7.1	<i>Requerimientos del prototipo</i>	48
2.7.2	<i>Diagrama de flujo del programa del módulo Lectura, Transmisión y Activación</i>	48
2.7.3	<i>Diagrama de flujo del programa del módulo Recolección, Visualización y Control...</i>	51
CAPITULO III		
3.	PRUEBAS Y RESULTADOS	55
3.1	Introducción	55
3.2	Caracterización del prototipo	55
3.3	Caracterización del módulo Recolección, Visualización y Control	56
3.4	Caracterización del módulo Lectura, Transmisión y Activación	56
3.5	Caracterización de la red local de agua	58
3.6	Caracterización del flujo de agua	59
3.6.1	<i>Repetitividad del Lectura, Transmisión y Activación</i>	59
3.6.2	<i>Calibración del sensor de flujo</i>	61
3.6.3	<i>Comprobación de la calibración del Sensor de flujo</i>	62
3.7	Prueba de la Comunicación del prototipo	66
3.7.1	<i>Prueba de transmisión de datos</i>	66
3.7.2	<i>Prueba de transmisión control a los nodos</i>	67
3.7.3	<i>Prueba de transmisión de alerta</i>	68
3.8	Prueba de comunicación con la base de datos local	69
3.9	Prueba de comunicación con la página web	70
3.10	Prueba de alimentación	71
3.11	Implementación del prototipo	72
3.12	Análisis económico del prototipo	73
CONCLUSIONES		76
RECOMENDACIONES		77
BIBLIOGRAFÍA		
ANEXOS		

ÍNDICE DE TABLAS

Tabla 1-1: Características de las redes de distribución	10
Tabla 2-1: Características de diferentes modelos de placas de Arduino.....	17
Tabla 3-1: Características de diferentes modelos de placas Raspberry Pi	19
Tabla 4-1: Características de eléctricas de la Intel galileo	20
Tabla 5-1: Características de diferentes lenguajes de programación	22
Tabla 1-2: Modelos de Arduino seleccionados.....	33
Tabla 2-2: Características del módulo Nrf24L01.....	35
Tabla 3-2: Características del módulo Sim800L.....	36
Tabla 4-2: Características del Sensor YF-S201	38
Tabla 5-2: Características de la electroválvula 12V DC	40
Tabla 6-2: Especificaciones módulo Tp4056.....	41
Tabla 7-2: Pines de conexión Arduino Nano y NRF24L01	45
Tabla 8-2: Pines de conexión Arduino Mega y NRF24L01	47
Tabla 9-2: Pines de conexión Arduino Mega y SIM 800L	47
Tabla 1-3: Prueba de repetitividad sensores :1,2,3	59
Tabla 2-3: Prueba de repetitividad sensores :4,5,6	60
Tabla 3-3: Estadística descriptiva para la repetitividad de monitoreo	61
Tabla 4-3: Calibración de los sensores en función de pulsos.....	62
Tabla 5-3: Pruebas del sensor 1, 2 ,3 después de la calibración	63
Tabla 6-3: Pruebas del sensor 4, 5 ,6 después de la calibración	63
Tabla 7-3: Estadística descriptiva del error después de la calibración.....	64
Tabla 8-3: Cadena de caracteres de información de censado	67

Tabla 9-3: Cadena de caracteres de información de control	68
Tabla 10-3: Configuración de para enlace con la base de datos	69
Tabla 11-3: Potencia de Consumo del prototipo.....	71
Tabla 12-3: Análisis económica del prototipo	74

ÍNDICE DE FIGURAS

Figura 1-1: Proceso de monitorización.....	6
Figura 2-1: Red de distribución con circuito abierto	8
Figura 3-1: Red de distribución con circuito cerrado	9
Figura 4-1: Red de sensores inalámbricos (WSN).....	11
Figura 5-1: Partes de un nodo sensor.....	12
Figura 6-1: Tipología de la red de sensores	13
Figura 7-1: Módulos Arduino.....	16
Figura 8-1: Placa Raspberry Pi	18
Figura 9-1: Placa Galileo	20
Figura 10-1: Sensores	21
Figura 11-1: Estructura de una DB	25
Figura 1-2: Concepción de arquitectura módulo Lectura, Transmisión y Activación.....	28
Figura 2-2: Concepción de arquitectura módulo Recolección, Visualización y Control.....	29
Figura 3-2: Arquitectura del módulo de Lectura, Transmisión y Activación	31
Figura 4-2: Arquitectura del módulo de Recolección, Visualización y Control.....	32
Figura 5-2: Arduino Nano.....	34
Figura 6-2: Arduino Mega.....	34
Figura 7-2: Módulo Nrf24L01	36
Figura 8-2: Módulo SIM800L	37

Figura 9-2: Módulo ethernet shield	38
Figura 10-2: Sensor de flujo YF-S201	39
Figura 11-2: Electroválvula 12V DC.....	41
Figura 12-2: Batería de litio de 3.7v y 1200mA	42
Figura 13-2: Módulo Tp4056	42
Figura 14-2: Panel solar 9v -100mA.....	43
Figura 15-2: Fuente de 900W	43
Figura 16-2: Esquema conexión del módulo Lectura, Transmisión y Activación.....	44
Figura 17-2: Esquema conexión del módulo Recolección, Visualización y Control	46
Figura 18-2: Diagrama de flujo del módulo Lectura, Transmisión y Activación.....	50
Figura 19-2: Diagrama de flujo del módulo Recolección, Visualización y Control .1.....	53
Figura 20-2: Diagrama de flujo del módulo Recolección, Visualización y Control .2.....	54
Figura 1-3: Fotografía del módulo Recolección, Visualización y Control	56
Figura 2-3: Fotografía del módulo Lectura, Transmisión y Activación (NODO 1)	57
Figura 3-3: Fotografía del módulo Lectura, Transmisión y Activación (NODO 2)	57
Figura 4-3: Fotografía del módulo Lectura, Transmisión y Activación (NODO 3)	57
Figura 5-3: Fotografía de la red local de agua potable	58
Figura 6-3: Prueba de transmisión de datos.....	66
Figura 7-3: Prueba de transmisión en el observado en el computador	67
Figura 8-3: Prueba de comunicación mediante GSM.....	68
Figura 9-3: Archivos de configuración de enlace a la base de datos	69
Figura 10-3: Fotografía de la comunicación entre la base de datos y Arduino.....	70

Figura 11-3: Pruebas de comunicación con la página web.....	70
Figura 12-3: Módulos antes de conectar a la red de agua potable	73
Figura 13-3: Implementación del prototipo en la red local de agua potable.....	73

ÍNDICE DE GRÁFICOS

Grafico 1-3: Pruebas de error de los sensores después de la calibración.....	65
---	----

ÍNDICE DE ABREVIATURAS

ACK	ACUSE DE RECIBO
PC:	COMPUTADOR PORTÁTIL
GPM:	GALONES POR MINUTO
M ³ :	METROS CÚBICOS
PA:	PASCALES
CTS	CLEAR TO SEND
SQL	STRUCTURED QUERY LANGUAGE
WWAN	WIRELESS WIDE ÁREA NETWORK
WSN	WIRELESS SENSOR NETWORK
IR	INFRARROJOS
MW	MICROONDAS
RF	RADIO FRECUENCIA
HTML	HYPertext MARKUP LANGUAGE
EEPROM	ELECTRICALLY ERASABLE PROGRAMMABLE ROM
GPS	GLOBAL POSITIONING SYSTEM
SPI	SERIAL PERIPHERAL INTERFACE
TCP	TRANSMISSION CONTROL PROTOCOL
WPAN	WIRELESS PERSONAL AREA NETWORK
WSN	WIRELESS SENSOR NETWORK

ÍNDICE DE ANEXOS

- Anexo A:** Modulo de comunicación NRF24L01
- Anexo B:** Modulo shield Ethernet de Arduino
- Anexo C:** Datasheet Arduino Mega
- Anexo D:** Datasheet Arduino Nano
- Anexo E:** Sensor de Flujo de Agua Efecto Hall Modelo YF-S201
- Anexo F:** Modulo de comunicación SIM800L
- Anexo G:** Código módulo Recolección, Visualización y Control
- Anexo H:** Código módulo Lectura, Transmisión y Activación
- Anexo I:** Código de calibración del sensor
- Anexo J:** Tabla de correlación
- Anexo K:** Código configuración Arduino y la base de datos
- Anexo L:** Código de enlace de datos Arduino y la base de datos
- Anexo M:** Fotografías del prototipo implementado
- Anexo N:** Página web implantada
- Anexo O:** Fotografía de la base de datos implantada

RESUMEN

Se implementó un sistema electrónico inteligente de Medición, Control, Visualización y Alerta de fallas en un circuito local de agua potable de bajo costo, fácil implementación el cual permite la monitorización por medio de una página web. Con el desarrollo del prototipo se buscó dar una solución rápida cuando en las tuberías de distribución de agua potable existan fallas, generando falta de servicio a los usuarios y pérdida de litros de agua hasta la solución del mismo. El prototipo permite agregar nuevos módulos para la obtención de datos y a su vez controlarlos. Posee dos módulos desarrollados en la plataforma Arduino, la comunicación entre los módulos es mediante radio frecuencia utilizando el módulo NRF24L01 con una distancia máxima de 40m de transmisión y recepción, el primer módulo denominado nodo, posee una electroválvula que permite la detención o paso de agua potable hacia el ramal que se encuentre controlado por el sistema. Para obtención de los datos se utilizó sensores de flujo YF-S201, estos datos se envían de forma ordenada hacia el segundo módulo, la información recibida posee un identificador que facilita el procesamiento con el fin de controlar cada nodo, enviando un mensaje de activación o desactivación de las electroválvulas. Además, envía un mensaje de texto a través del sistema global para las comunicaciones móviles (GSM) cuando exista alguna falla en los ramales. Los datos procesados son almacenados en una base de datos un tiempo establecido. De las pruebas realizadas se comprobó que el dispositivo posee un error de monitoreo mínimo que es despreciable en cada uno de los sensores. Cada medición se hizo en comparación del volumen de agua. Se concluye que el prototipo permite un rápido control ante las fallas producidas en un circuito de agua potable dando estabilidad del servicio a los usuarios. Se recomienda para evitar errores de obtención de datos de flujo siempre calibrar los sensores.

Palabras claves: <TECNOLOGÍA Y CIENCIAS DE LA INGENIERÍA>, <TECNOLOGÍA DEL CONTROL AUTOMÁTICO>, <AGUA POTABLE>, <COMUNICACIONES INALÁMBRICAS >, <ARDUINO (HARDWARE - SOFTWARE)>, <SENSOR DE FLUJO>, <DISEÑO MODULAR>, <CONTROL DE FALLAS>.

ABSTRACT

It was implemented an intelligent electronic system for Measurement, Control, Visualization and Fault Alerting in local circuit of low cost drinking water, easy implementation which allows the monitoring through a web page. The development of the prototype is an attempt to provide quick solution, since there are faults in the potable water distribution pipes. It generates a lack of service to user and loss of liters of the water until the solution of the same. The prototype allows adding new modules of obtaining data and control. It has two modules developed in the Arduino platform. The communication between the modules is by radio frequency using the module NRFL24L01 with a maximum distance of 40m of transition and reception. The first module called node, has an electrovalve that allows the stop or passage of water to the branch that is controlled by the system. Flow sensor YF-S201 were used to obtain the data. These data are sent in an orderly way to the second module, the information received has an identifier that facilitates the processing in order to control each node it sends an activation message or deactivation of the solenoid valves. In addition, it sends a text message through the global system for mobile communications (GSM) when there is failure in the branches. The processed data are stored in a database for a set time. From the tests performed it is verified that the device has a minimum census error that is negligible in each of the sensor. Each measurement was made in comparison to the volume water. It is concluded that the prototype allows a fast a control of the produced faults in a potable water circuit, which gives a stability of the service to the users. It is recommended to avoid error in obtaining flow data through always calibrating the sensor.

KEYWORDS :< ENGINEERS TECHNOLOGY AND SCIENCES>, <AUTOMATIC CONTROL TECHNOLOGY>, <DRINKING WATER>, <WIRELESS COMMUNICATIONS>, <ARDUINO (HARDWARE-SOFTWARE)>, <FLOW SENSOR>, <MODULAR DESIGN>, <TROUBLESHOOTING>.

INTRODUCCIÓN

En el presente trabajo de titulación, se realizó la implementación de un prototipo electrónico para la Medición, Visualización, Control de averías, fallas de agua potable, evitando el desperdicio de líquido vital en una red de agua potable, logrando estabilidad y confianza por la empresa que proporciona este servicio.

En la actualidad no existe un dispositivo que permita determinar de forma oportuna fallas o averías en la tubería de agua potable en un circuito local dentro de una red. De esta manera se disminuye las pérdidas y desperdicios de agua dificultando el suministro normal de líquido vital a los usuarios de la red.

Para el cumplimiento de este objetivo, se implementó un prototipo el cual permite censar el flujo de agua que atraviesa por una tubería. Los actuadores instalados en el prototipo permiten abrir o cerrar las electroválvulas permitiendo o no el paso de líquido vital, los datos obtenidos por medio del sensor de flujo YF-S201 son enviados mediante Radio Frecuencia al módulo Recolección, Visualización y Control creando una WSN entre este y el módulo de Lectura, Transmisión y Activación. El prototipo a su vez permite visualizar el comportamiento del sistema por medio de una página web que es utilizada por el personal asignado.

Las redes de sensores inalámbricos (WSN) permiten una comunicación, por ser dispositivos autónomos permiten censar cualquier variable física o química existente y son de bajo costo.

En la mayoría de los casos, para solventar un problema de falla en las tuberías de distribución de agua potable no se lo realiza de forma inmediata, debido a que no se dispone de un sistema en tiempo real que alerte del problema suscitado en la red de agua potable.

Esta necesidad motiva el desarrollo de un prototipo de bajo costo que permita el monitoreo y control de fallas de agua potable en la tubería, de tal forma que la información suministrada pueda ayudar tanto a la empresa que suministra el servicio como a los usuarios de servicio, a tener una mejor estabilidad de suministro de líquido vital, evitando desperdicios de agua.

Este trabajo de titulación consta de una introducción, tres capítulos, conclusiones y recomendaciones. El primer capítulo, denominado Marco Teórico contiene el estudio de los métodos de distribución de agua potable para la población, una breve revisión de los elementos que conforman el prototipo de Medición, Visualización y Control de falla en una red de agua potable.

El segundo capítulo es el Marco Metodológico plantea todos los requerimientos que tiene el sistema para la selección de los dispositivos, a su vez las características de cada dispositivo que permitieron su elección. Presenta el hardware y software utilizados para la implementación del módulo Lectura, Transmisión y Activación y del módulo Recolección, Visualización y Control.

El tercer capítulo analiza los resultados de las pruebas desarrolladas para cada dispositivo utilizado en cada módulo que conforma el prototipo, se incluye las características del sistema, así como el análisis económico de su costo de implementación.

En las Conclusiones se resumen los resultados alcanzados en el trabajo y se resaltan los aspectos importantes, en las Recomendaciones se citan los aspectos más importantes a ser considerados en el prototipo para un mejor manejo y vida útil.

ANTECEDENTES

Durante la última década han aparecido infinidad de dispositivos que utilizan sensores para el monitoreo y control. La evolución en prestaciones y tamaño que nos ofrece la electrónica actual, disponemos de sensores capaces de comunicarse de manera inalámbrica, capacidad de procesamiento y autonomía propia, abriendo un abanico de oportunidades para diseñar y crear todo tipo de aplicaciones, protocolos y sistemas capaces de facilitar el trabajo a los seres humanos a la vez que reducen sus costes. (Cantillo,2010, p.8)

En la actualidad las funciones que poseen las redes de sensores están donde se requiera recoger lecturas. Estas pueden ser en un entorno inaccesible u hostil como también en empresas, fabricas, oficinas u el propio hogar, durante un período de tiempo, para así detectar cambios, tendencias y datos suficientes para poder generar algún cambio o intervención. (TAPIA,2008, p.8)

Cada vez que se produce una fuga de agua, por pequeña que sea, en una de estas redes de abastecimiento, la pérdida recae sobre el conjunto de la población, por eso es necesario un control exhaustivo de estas instalaciones, porque suelen ser fuente de pérdidas económicas importantes sobre las que nadie suele responsabilizarse. (GÓMEZ,2015). Según la EP EMAPAR existe un total de 46 a 47 % de fugas de agua que posee la ciudad de Riobamba en sus tuberías de distribución.

Las redes de tubería de abastecimiento para hogares de líquido vital que se encuentran en la ciudad de Riobamba no son monitoreadas de forma que la empresa de distribución de agua potable, no sabe cuándo existe una ruptura de tuberías en los sectores que esta abastece con el líquido vital, lo cual produce pérdidas económicas para la empresa como a la población, lo cual no generando una estabilidad del servicio para los usuarios.

De las investigaciones realizadas no se encuentra en el mercado ningún dispositivo electrónico que realice un monitoreo del suministro de agua en una malla de distribución por parte de la empresa encargada, por lo que se ha previsto la realización de un sistema electrónico prototipo que venga a cumplir estas funciones.

FORMULACIÓN DEL PROBLEMA

¿Se podrá detectar y solucionar oportunamente los inconvenientes de suministro de agua en un circuito de distribución mediante la implementación de un dispositivo prototipo de monitoreo con comunicación inalámbrica, garantizando un mejor servicio a los usuarios?

SISTEMATIZACIÓN DEL PROBLEMA

¿Existe un dispositivo electrónico que permita monitorear un circuito de agua potable?

¿Cómo se realizará el monitoreo de un circuito de agua potable?

¿Cuáles serán los mecanismos de conectividad y alertas en caso ocurrir fallas en un circuito de agua potable?

¿Cómo se podrían detectar falla de funcionamiento en un circuito de agua potable para su rápida reparación?

OBJETIVOS

Objetivo general

Diseñar e implementar un prototipo mediante comunicación inalámbrica para el monitoreo de falla en un circuito de agua potable.

Objetivos específicos

- Analizar los elementos electrónicos y de comunicación para el sistema propuesto.
- Medir el flujo de líquido vital en un circuito de agua potable.
- Implementar una página web para el monitoreo de un circuito de agua.
- Implementar comunicación GSM para el envío de un mensaje de alerta y radio frecuencia para la conectividad de los nodos con el punto base.
- Implementar una electroválvula para el control del circuito de agua potable.

CAPITULO I

1. MARCO TEÓRICO REFERENCIAL

1.1 Agua

El agua es indispensable para la vida, porque ningún organismo sobrevive sin ella. Es un constituyente esencial de la materia viva y la fuente de hidrógeno para los organismos. También influye en ellos a través de la atmósfera y el clima. Es el medio en el que se desarrolla la abundante y variada flora y fauna acuática. Los seres vivos están formados en su mayor parte por agua.

El agua dulce es imprescindible para la vida, pero la cantidad disponible es escasa y su distribución desigual. Además, varía a lo largo del año y está sujeta a cambios provocados por la actividad humana. Los usos más importantes están relacionados con la agricultura y el consumo industrial y doméstico. Su demanda se ha incrementado notablemente con el crecimiento de la población. (FAO,2015)

La Tierra, con sus diversas y abundantes formas de vida, que incluyen a más de 6.000 millones de seres humanos, se enfrenta en este comienzo del siglo veintiuno con una grave crisis del agua. Todas las señales parecen indicar que la crisis se está empeorando y que continuará haciéndolo, a no ser que se emprenda una acción correctiva. Se trata de una crisis de gestión de los recursos hídricos, esencialmente causada por la utilización de métodos inadecuados. (UNESCO,2003)

1.2 Monitorización

La monitorización juega un papel importante en las actividades de calidad y gestión. El dato obtenido en una monitorización nos permite conocer el comportamiento y planificar una posible actuación, por lo que la información analizada es para optimizar la operación.

La monitorización consiste en la grabación continua, periódica o periódicamente continua de unos parámetros que, en la mejor manera explican el estado actual del sistema. La monitorización depende de varios factores como los materiales, condiciones ambientales y fenómenos locales. (SIDOROVA,2010, p.18-19)

1.2.1 Proceso de monitorización

Las actividades principales del proceso de monitorización son: selección de una estrategia de monitorización; instalación de un sistema de monitorización; mantenimiento del sistema de monitorización; colección y análisis de los datos obtenidos.

La a monitorización está basada en el conocimiento del sistema observado; la determinación del sistema de monitorización depende de: objetivo, parámetros seleccionados, y tipo de monitorización; exactitud, frecuencia de recepción de datos, manera de colección de datos, procedimiento de componentes del sistema, posibilidad de funcionamiento automático.se puede observar las actividades principales de monitorización en la figura 1-1. (SIDOROVA,2010, p.21)



Figura 1-1: Proceso de monitorización.

Fuente: (SALGADO, y otros, 2016, P. 5)

1.2.2 Monitorización del agua potable

El objetivo de la monitorización de la calidad de diferentes aguas naturales y potables es la reducción de los efectos nocivos para el medioambiente y la salud humana. Este objetivo tiene que cumplir con diferentes normas que están estipuladas en regulaciones oficiales sobre ecología y la calidad del agua potable.

Debido a los esfuerzos realizados continuamente para mejorar la calidad de las aguas naturales, reducir los riesgos en la salud de los consumidores y optimizar la eficacia del tratamiento del agua potable, continuamente aumentan los requisitos en los procesos tecnológicos y en el control de la calidad del agua. (S::CAN, 2017)

Por lo que es necesario ubicar una estación de monitorización inalámbrica para el control del suministro de agua potable.

1.3 Redes de distribución de agua

El abastecimiento de agua potable para localidades urbanas en una de sus últimas etapas lo conforman una serie de conexiones y tuberías que se instalan o localizan dentro de las redes de calles. La planta de las calles, la topografía y la localización de las obras de cabeza, nos determinan el tipo de distribución y el flujo a través de aquel. (OZUNA, y otros, 1991)

1.3.1 Estructura jerárquica

Las redes de distribución de agua se estructuran según el tipo de función que desempeñan y que tienen una relación directa con la serie decreciente de los diámetros con el fin de ajustarse a la distribución de consumos, a la reducción de pérdidas de carga, hacer frente a situaciones imprevistas y a reducir el coste. Los niveles en los que se clasifican son: Aducción, Distribución Urbana y acometidas.

Las conducciones de aducción son las grandes arterias de transporte que recogen el agua desde los puntos de captación o tratamiento y llegan hasta la zona urbana. La red de distribución urbana es la que toma el agua de la aducción directamente o de los diferentes depósitos reguladores y la distribuye entre todos los puntos de consumo. Dentro de esta red se ha de distinguir las redes de transporte inter-zonales. (MOLIA,2010, p.3)

1.3.2 Tipologías

Las redes de distribución en general o bien según su función o localización por áreas pueden ser de dos grandes tipos: ramificada y mallada

1.3.2.1 Red ramificada o circuito abierto

Cuando los asentamientos de una localidad son irregulares y la traza urbana no sigue alguna planeación adecuada y por lo general existe una sola calle o avenidas, callejones en diferentes direcciones, en estas localidades es común la instalación de sistemas de distribución de “ramales abiertos”, en donde los extremos de tubería serán puntos muertos y el agua no tendrá circulación. Como se puede observar en la figura 2-1.

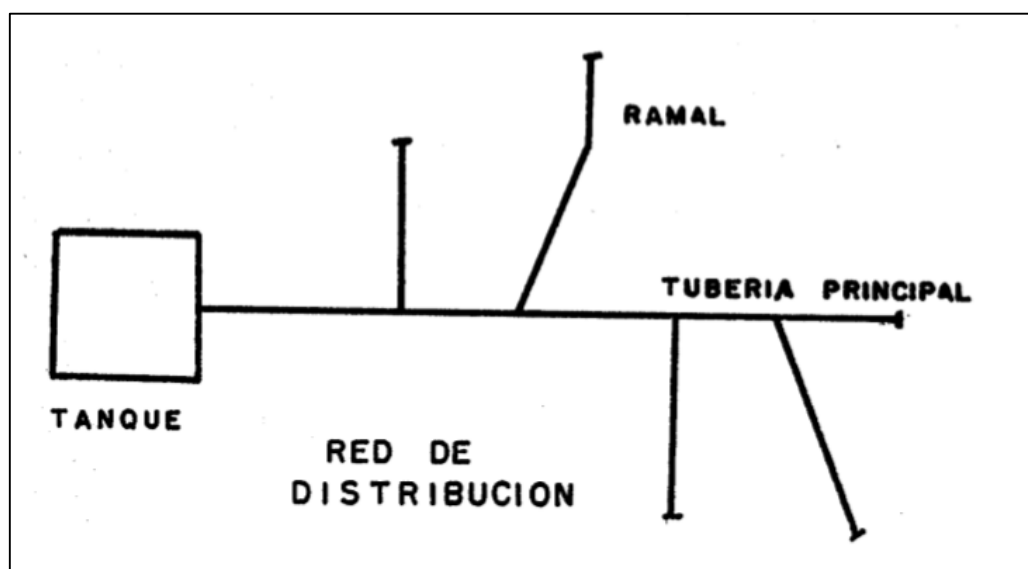


Figura 2-1: Red de distribución con circuito abierto

Fuente: (ROMO, 1991)

1.3.2.2 Red mallada o circuito cerrado

Cuando se presenta una adecuada distribución de calles y avenidas existiendo una cuadrícula bien definida de estas, se procederá a construir un sistema de red de distribución con “circuitos principales” o “circuitos cerrados”, en donde estos alimentan a líneas de tubería, denominadas de relleno, las cuales en los cruces de calles se interconectarán entre sí, o con las tuberías principales o de alimentación.

Este sistema presentará la ventaja de que la alimentación podrá realizarse desde cualquier punto y desde una más dirección como se observa en la figura 3-1, eliminando así los extremos muertos para dar mejor funcionalidad hidráulica a la red. (OZUNA, y otros, 1991)

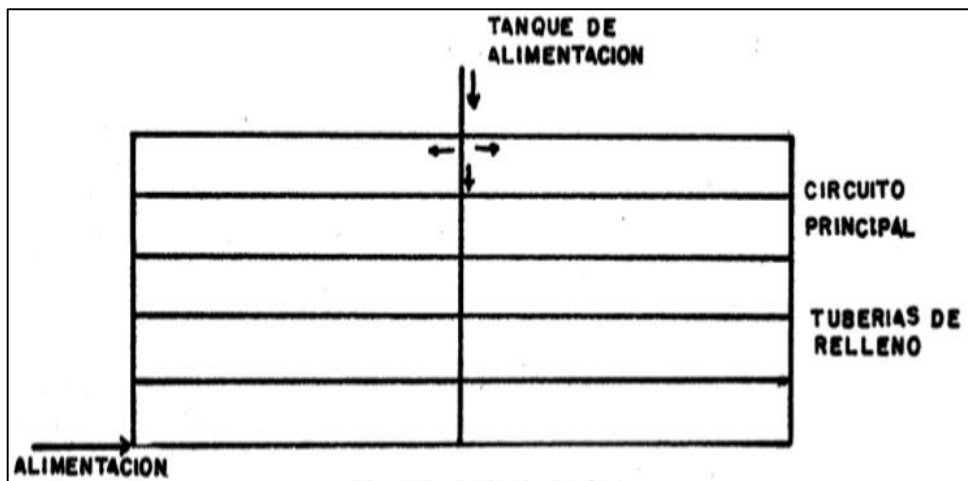


Figura 3-1: Red de distribución con circuito cerrado

Fuente: (ROMO, 1991)

1.3.2.3 Características de las redes de distribución de agua potable

Las redes de distribución de agua potable poseen características las cuales le permiten abastecer de agua a los diferentes puntos de una población.

La tabla 1-1 indica las características de las redes de distribución de agua potable y sus diferentes delimitaciones que posee cada una.

Tabla 1-1: Características de las redes de distribución

	Ramificada	Mallada
Redes de distribución de agua potable	<ul style="list-style-type: none"> • Núcleos urbanos de máximo 1000 viviendas. • Configuración urbana lineal. • Longitud máxima de las arterias hasta 1000. • Los distribuidores tendrán una longitud máxima de 300m. • Una ruptura puede generar el corte general de la distribución. • Menor coste de montaje de la red de agua 	<ul style="list-style-type: none"> • La separación máxima de los lados opuestos será de 900m y mínima de 250m. • La superficie máxima es de 30Ha y mínima de 9Ha. • Núcleos urbanos máxima de 1500 viviendas. • Mayor seguridad en el servicio, ya que una avería en punto no acarrea a toda la red. • Mayor coste de montaje de la red de agua

Fuente: ACUEDUCTO, 2008

Realizado por : GUAÑO Wilmer, 2017

1.4 Red de sensores inalámbricos (WSN)

La red de sensores viene de la lengua inglesa Wireless Sensor Networks o WSN se basa en una serie de pequeños dispositivos electrónicos que tienen acceso al mundo exterior por medio de sensores.

El nombre que se le da a este tipo de dispositivo es el de “mote”, que proviene de la tradición inglesa de la palabra “mota de polvo” que proviene con la finalidad de indicar en una sola palabra dos de los conceptos principales: su pequeño tamaño y la idea de que pueden estar situados en cualquier lugar. Esta facilidad en su instalación y funcionamiento la consiguen principalmente a dos motivos:

- Son dispositivos autónomos que funcionan con baterías (similares a la de un teléfono móvil) las cuales permiten ser recargadas incluso mediante placas solares.
- Todas las comunicaciones que se realizan se basa en protocolos inalámbricos de bajo consumo.

Los “motes” tienen la capacidad de comunicarse entre sí mediante la creación de redes malladas y retransmitir la información requerida a través de la red hasta un punto de control que registre los valores observados e incluso tome decisiones consecuentes.

Son los propios “motes” los que se organizan cada cierto tiempo para ver cuáles son las rutas de comunicación disponibles. Esto permite que los motes puedan ser cambiados de lugar para monitorizar un área distinta en un determinado momento. (GASCÓN,2010, p. 53) La figura 4-1 indica la función de cada elemento que interviene en la WSN.

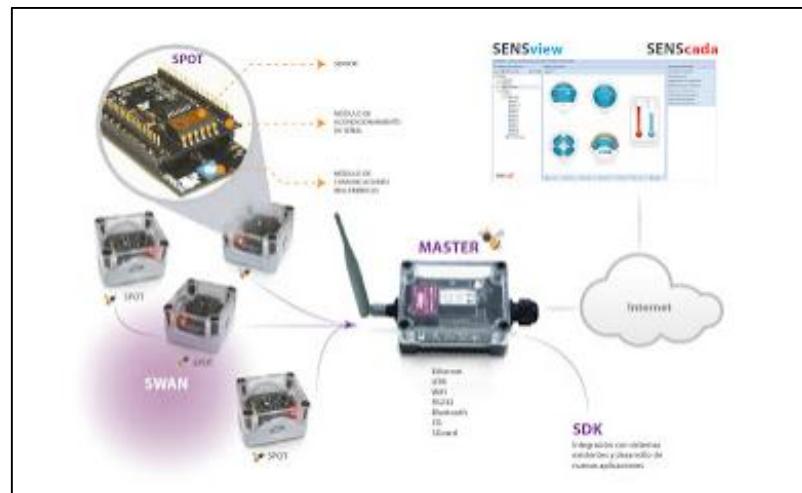


Figura 4-1: Red de sensores inalámbricos (WSN)

Fuente: http://lasredesensensores.blogspot.com/2012_07_01_archive.html

1.4.1 Hardware de un nodo sensor

Los nodos inalámbricos o motes son dispositivos electrónicos capaces de captar información proveniente del entorno en el que se encuentran, procesarla y transmitirla inalámbricamente hacia otro destinatario.

Diseñar una mota no se reduce a miniaturizar un ordenador personal. Hay que tener en cuenta que queremos un espacio reducido, un consumo muy bajo de energía y un coste de los dispositivos reducido. Y en contraposición a esto una potencia de ejecución de programas elevadas y una transmisión de datos eficaz y con amplia longitud de emisión. (Cantillo, 2010, p.16)

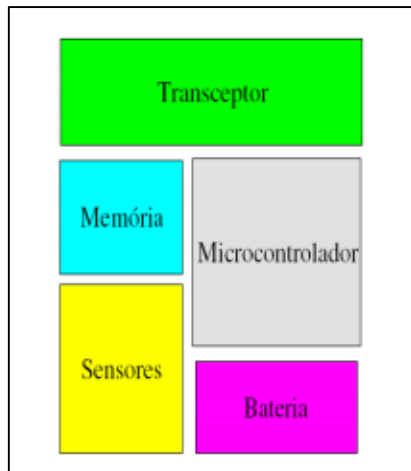


Figura 5-1: Partes de un nodo sensor

Fuente: [https://riunet.upv.es/bitstream/handle/10251/8592/PFC%20%20DESARROLLO%](https://riunet.upv.es/bitstream/handle/10251/8592/PFC%20%20DESARROLLO%20)

Un nodo sensor es un elemento computacional con capacidad de procesamiento, memoria, interfaz de comunicación y puede formar conjuntos de sensores.

El Hardware básico de un nodo sensor como se observa en la figura 5-1 se compone de un transceptor (transmisor/receptor), procesador, uno o más sensores, memoria y batería. Los componentes brindan la opción de comunicación (enviar/recibir información), ejecutar tareas que requieren procesamiento más allá de efectuar funciones de censado.

1.4.2 Tipologías con la red con sensores

Se usa varias topologías de red para coordinar el gateway WSN, nodos finales u otros ruteadores. Los nodos ruteadores son similares a los nodos finales, en los que se puede adquirir datos de medida, pero usted también puede usarlos para pasar datos de medida desde otros nodos. (CIANCA,2012)

- **La topología en estrella**, en la cual cada nodo mantiene una sola trayectoria de comunicación directa con el gateway. Esta topología es simple, pero restringe la distancia total que su red puede alcanzar
- **Topología de cluster o árbol**, esta arquitectura más compleja, cada nodo mantiene una sola trayectoria de comunicación directa al gateway pero puede usar otros nodos para enrutar sus datos a esa trayectoria. Esta topología tiene un problema. Si el nodo ruteador

falla, todos los nodos que dependen de ese ruteador también pierden sus trayectorias de comunicación con el gateway.

- **La topología de red de malla**, soluciona los problemas con las trayectorias de comunicación redundante para aumentar la fiabilidad del sistema, los nodos mantienen múltiples trayectorias de comunicación al gateway, así si el nodo ruteador falla, la red automáticamente en ruta los datos a través de una trayectoria diferente, aunque es muy confiable, sufre de incremento de latencia de red ya que los datos deben hacer múltiples saltos antes de llegar al gateway.

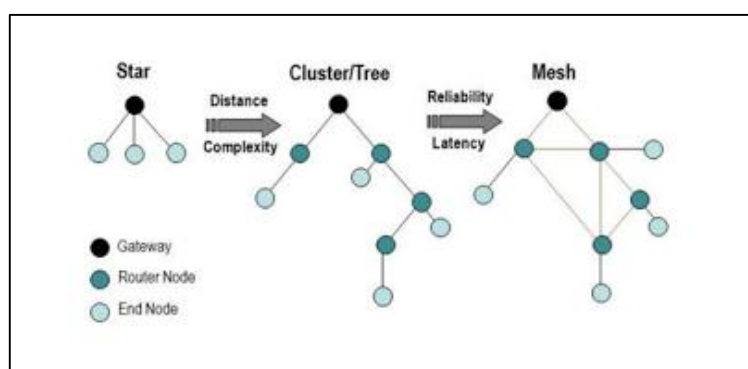


Figura 6-1: Tipología de la red de sensores

Fuente: http://lasredesconsensores.blogspot.com/2012_07_01_archive.html

Cada tipología como se observa en la figura 6-1 está diseñada para cada situación que se requiera el uso de ella.

1.4.3 Áreas de aplicación

Pasando de largo las aplicaciones militares, éstas tienen usos civiles interesantes como vemos las cuales satisfacen necesidades que el hombre necesita en su vida cotidiana continuación: (CIANCA,2012)

- **Eficiencia energética:** Red de sensores se utilizan para controlar el uso eficaz de la electricidad, como el caso de Japón y España.

- **Sensores ambientales:** El control ambiental de vastas áreas de bosque o de océano, sería imposible sin las redes de sensores. El control de múltiples variables, como temperatura, humedad, fuego, actividad sísmica, así como otras. También ayudan a expertos a diagnosticar o prevenir un problema o urgencia y además minimiza el impacto ambiental.
- **Sensores industriales:** Dentro de fábricas existen complejos sistemas de control de calidad, el tamaño de estos sensores les permite estar allí donde se requiera.
- **Automoción:** Las redes de sensores son el complemento ideal a las cámaras de tráfico, ya que pueden informar de la situación del tráfico en ángulos muertos que no cubren las cámaras y también pueden informar a conductores de la situación, en caso de atasco o accidente, con lo que estos tienen capacidad de reacción para tomar rutas alternativas.
- **Medicina:** Las redes de sensores ayudan a la calidad de vida de pacientes que deben estar controlado, sus constantes vitales (pulsaciones, presión, nivel de azúcar en sangre, etc.), podrá mejorar sustancialmente.
- **Domótica:** Su tamaño, economía y velocidad de despliegue, lo hacen una tecnología ideal para la domótica a un precio asequible.

El futuro de la era de la información es capaz de digitalizar en tiempo real el mayor número de posibles parámetros del entorno con la idea de controlar y optimizar el uso que se hace de sus recursos mejorando con ello la calidad de vida de las personas que viven en ellos. (GASCÓN,2010, p. 55)

A partir de ahí se ha ido investigando sobre las diferentes formas de obtener información en los diferentes ámbitos, lo que ha llevado a que los sensores evolucionen hasta el momento con las redes de sensores inalámbricas, las cuales son capaces de obtener información periódica en las diferentes facetas sin tener que estar una persona presente en el medio físico y exentas de mantenimiento.

Se crean redes de sensores que constan de una cantidad importante de nodos autónomos capaces de auto configurarse en el caso de que alguno de ellos deje de funcionar y elegir un camino alternativo a la hora de propagar la información hasta el sumidero para su posterior envío por Internet hasta los equipos de monitorización. (Cantillo,2010, p.12)

1.5 Comunicación inalámbrica

Actualmente, estamos inmersos en la que se denomina revolución tecnológica de las comunicaciones inalámbricas, una revolución similar a la que protagonizaron en su momento la electricidad, la televisión, el ordenador o las mismas comunicaciones con cable, que supusieron nuevos modelos de negocio.

1.5.1 Clasificación por rangos de frecuencia

Una de las principales ventajas de esta tecnología es la movilidad, no depender del cable. El hecho de que el punto de entrada en la red de comunicaciones no esté ligado a una ubicación fija, El medio de transmisión de las comunicaciones inalámbricas (no guiadas) es el espectro electromagnético que coloquialmente denominamos aire.

Los rangos de frecuencia más utilizados que se utiliza para la comunicación inalámbrica son los siguientes:

1.5.1.1 Infrarrojos (IR)

Se utilizan en comunicaciones punto a punto de corto alcance, son muy direccionales y no pueden atravesar obstáculos. Este medio se utiliza habitualmente en el mando a distancia de la televisión y hasta hace unos años era también un sistema de comunicación que se utilizaba a menudo para conectar dispositivos situados el uno al lado del otro. Es el rango de frecuencia más alto para comunicaciones inalámbricas. (BLAZQUEZ,2013)

1.5.1.2 Microondas (MW)

Este rango de frecuencias es adecuado para transmisiones de largo recorrido (comunicaciones por satélite, comunicaciones terrestres punto a punto como alternativa al cable coaxial o la fibra óptica. Las microondas suelen ser direccionales y utilizan una parte del espectro con frecuencias más pequeñas que los infrarrojos. (BLAZQUEZ,2013) Las microondas están dentro de una gama de frecuencia de 300 MHz a 300 GHz.

1.5.1.3 Radiofrecuencias (RF).

Las radiofrecuencias son omnidireccionales y pueden atravesar obstáculos sin ningún problema. (BLAZQUEZ,2013) Las radiofrecuencias oscilan entre 10 kHz y 300 GHz Las microondas están incluidas dentro de la banda de radiofrecuencia. (E DOME, 2017)

1.6 Tarjetas de desarrollo

Dentro de las tarjetas de desarrollo tenemos diferentes tipos las cuales poseen características que las definen para su respectivo funcionamiento.

1.6.1 Arduino

Arduino es una plataforma de hardware y software de código abierto, basada en una sencilla placa con entradas y salidas, analógicas y digitales, en un entorno de desarrollo que está basado en el lenguaje de programación Processing. Al ser open source, tanto su diseño como su distribución, puede utilizarse libremente para el desarrollo de cualquier tipo de proyecto sin necesidad de licencia. (AMANGANDI,2012)

Con el pasar del tiempo fueron desarrollándose diferentes modelos de Arduino como se ve en la figura 7-1 y sus características en la tabla 2-1

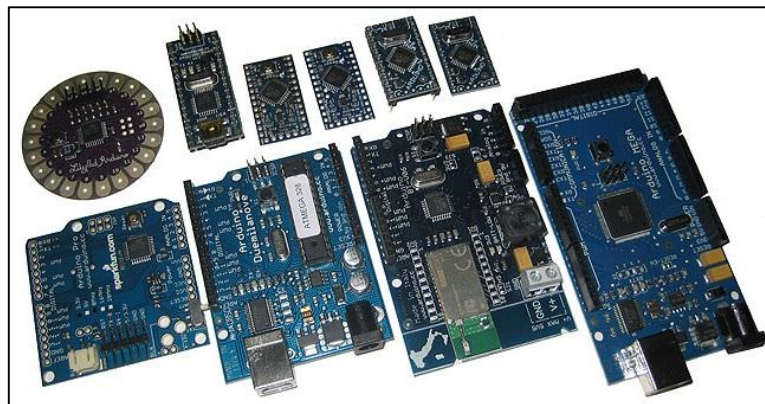


Figura 7-1: Módulos Arduino

Fuente: <http://recursos.cepindalo.es/mod/book/tool/print/index.php?id=2886>

Tabla 2-1: Características de diferentes modelos de placas de Arduino

Modelo	Voltaje de entrada	Frecuencia de reloj	E/S digitales	E analógicas	PWM
Arduino Due	5-12V	84MHz	54*	12	12
Arduino Leonardo	7-12V	16MHz	20*	12	7
Arduino Uno -R3	7-12V	16MHz	14	6	6
Arduino Bluetooth	7-15V	16MHz	14	6	6
RedBoard	1.2-5,5V	16MHz	14	6	6
Arduino Pro 3.3v/ 8MHz	3.35 -12V	8MHz	14	6	6
Arduino Pro 5v/ 16MHz	5 - 12V	16MHz	14	6	6
Arduino Mega 2560 R3	7-12V	16MHz	54	16	14
Mega Pro 3.3v	3.3-12V	8MHz	54	16	14
Mega Pro 5v	5-12V	16MHz	54	16	14
Arduino Mini 05	7-9V	16MHz	14	6	8
Arduino Pro Mini	3.35-12V	8MHz	14	6	6
Arduino Pro Mini	5 - 12V	16MHz	14	6	6
Arduino Fio	3.35-12V	8MHz	14	8	6
Mega Pro Mini 3.3V	3.35-12V	8MHz	54	16	14
Pro Micro 5V/16MHz	5-12V	16MHz	12	4	5
Pro Micro 3.3V/8MHz	3.35-12V	8MHz	12	4	5
LilyPad Arduino 328	2.7-5,5V	8MHz	14	6	6
LilyPad Arduino	2.7-5,5V	8MHz	9	5	5

Fuente: <https://es.wikipedia.org/wiki/Arduino>

Realizado por : GUÑO Wilmer, 2017

1.6.1.1 Shield

Un shield es una placa impresa que se pueden conectar en la parte superior de la placa Arduino para ampliar sus capacidades. Son diseños bastante simples y en general de código abierto, los diseños suelen ser publicados libremente. (CESAR,2008)

1.7 Raspberry Pi

El Raspberry Pi es un micro ordenador o una placa de computadora de bajo costo (figura 8-1) desarrollada en el Reino Unido por la Fundación Raspberry Pi, de manera que pueda fomentar la enseñanza de la computación y programación en las escuelas y colocarla al alcance de todos.

Entre los sistemas operativos disponibles para Raspberry Pi se encuentran: Raspbian, Arch Linux, RaspBMC, Pidora u OpenELEC e incluso, se dispone de Windows 10 según la página web de Microsoft. (CULTURACION,2016)

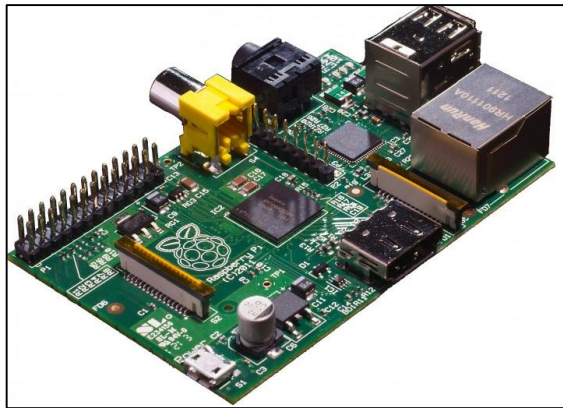


Figura 8-1: Placa Raspberry Pi

Fuente: <http://culturacion.com/raspberry-pi-que-es-caracteristicas-y-precios/>

El concepto es el de un ordenador desnudo de todos los accesorios que se pueden eliminar sin que afecte al funcionamiento básico. Está formada por una placa que soporta varios componentes necesarios en un ordenador común y es capaz de comportarse como tal.

A la Raspberry Pi la han definido como una maravilla en miniatura, que guarda en su interior un importante poder de cómputo en un tamaño muy reducido. Es capaz de realizar cosas extraordinarias (CONTRERAS, 2013)

Existe diferentes modelos de Raspberry pi que en la tabla 3-1 se presenta.

Tabla 3-1: Características de diferentes modelos de placas Raspberry Pi

	Model A	Model A+	Model B	Model B+	2 Model B	Zero	3 Model B	Zero W
SoC	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2835	Broadcom BCM2836	Broadcom BCM2835	Broadcom BCM2837	Broadcom BCM2835
CPU	700MHz ARM1176JZF-S	700MHz ARM1176JZF-S	700MHz ARM1176JZF-S	700MHz ARM1176JZF-S	900MHz Quad-core ARM Cortex-A7	1GHz ARM1176JZF-S	1.2GHz QUAD ARM Cortex-A53	1GHz ARM1176JZF-S
GPU	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV	VideoCore IV
RAM	256Mb	256Mb	512Mb	512Mb	1Gb	512Mb	1Gb	512Mb
USB	1	1	2	4	4	1 Micro	4	1 Micro
Vídeo	RCA, HDMI	Jack, HDMI	RCA, HDMI	Jack, HDMI	Jack, HDMI	Mini HDMI	Jack, HDMI	Mini HDMI
Audio	Jack, HDMI	Jack, HDMI	Jack, HDMI	Jack, HDMI	Jack, HDMI	Mini HDMI	Jack, HDMI	Mini HDMI
Boot	SD	MicroSD	SD	MicroSD	MicroSD	MicroSD	MicroSD	MicroSD
Red	-	-	Ethernet 10/100	Ethernet 10/100	Ethernet 10/100	-	Ethernet 10/100, Wifi, BT	Wifi y BT
Consumo	300mA / 1,5w / 5v	400mA / 2w / 5v	700mA / 3,5w / 5v	500mA / 2,5w / 5v	800mA / 4w / 5v	160mA / 0,8w / 5v	2,5A / 12,5w / 5v	160mA / 0,8w / 5v
Alimentación	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO	MicroUSB / GPIO
Tamaño	85,6 x 53,98 mm	65 x 56 mm	85,6 x 53,98 mm	85 x 56 mm	85 x 56 mm	65 x 30 mm	85 x 56 mm	65 x 30 mm

Fuente: <https://i0.wp.com/comohacer.eu/wp-content/uploads/comparativa-raspberry-pi-2.png>

Realizado por : <https://i0.wp.com/comohacer.eu/wp-content/uploads/comparativa-raspberry-pi-2.png>

1.7.1 Galileo

El Galileo es una placa de desarrollo de Intel (figura 9-1) que sirve para la rápida creación de prototipos simples con diseños interactivos, como pantallas de luz que responden a social media o a cualquier otra fuente que el desarrollador principiante desee. O también para que los más avanzados puedan desarrollar proyectos complejos en donde la automatización de electrodomésticos. (VARONAS, 2013)

La placa Intel Galileo es la primera de una gama de placas de desarrollo compatibles con Arduino basadas en arquitecturas Intel. Esta placa de desarrollo ejecuta un sistema operativo Linux libre que contiene las librerías de software de Arduino,

Permite ofrecer una mayor escalabilidad y reutilizar el software ya existente, llamados “*bocetos*”. (MTURANA, 2013)

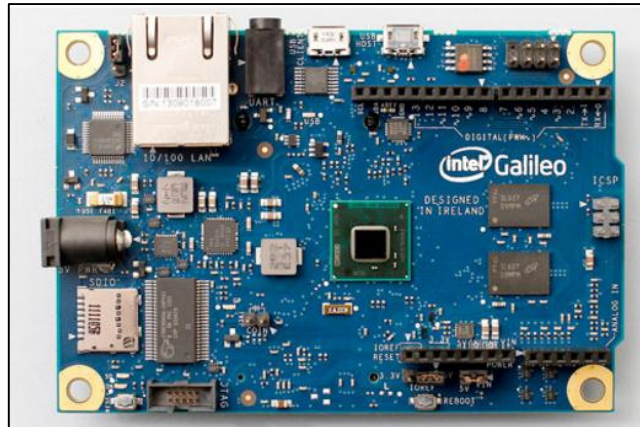


Figura 9-1: Placa Galileo

Fuente: <http://es.engadget.com/2013/10/03/intel-galileo-placa-desarrollo-arduino/>

La placa Galileo es también compatible en software con el Entorno de Desarrollo Integrado de Arduino, lo cual facilita la usabilidad e introducción. En adición a la compatibilidad en hardware y software con Arduino, la placa Galileo tiene varios puertos de E/S PC industrial estándar y características para expandir capacidades y uso nativo más allá del ecosistema de los shields para Arduino. (ANGELOU,2017). La tabla 4-1 especifica las características que posee esta placa de desarrollo.

Tabla 4-1: Características de eléctricas de la Intel galileo

Características eléctricas	
Voltaje de Entrada (recomendado):	5V
Voltaje de Entrada (límite)	5V
Pines de E/S Digital	14 (de los cuales 6 proveen salida PWM)
Pines de Entrada Análoga:	6
Corriente DC Total de Salida en todos los pines de E/S	80 mA
Corriente DC para pin de 3.3V	800 mA
Corriente DC para pin de 5V:	800 mA

Fuente: <http://tecbolivia.com/index.php/venta-de-componentes-electronicos-11/arduino/intel-galileo-detail>
Realizado por : GUANO Wilmer, 2017

Con las características que presentan cada placa de desarrollo, se escoge Arduino por estar enfoca a control de dispositivos que se requiere para el prototipo y su bajo costo. La placa de

desarrollo Raspberry pi está orientado para audio, video, alto costo de adquisición .la placa desarrollo Galileo es una plataforma que facilita la integración con Arduino, pero su costo es muy elevado para este prototipo.

1.8 Sensores

Un sensor consta de algún elemento sensible a una magnitud física como la intensidad o color de la luz, temperatura, presión, magnetismo, humedad, por su propia característica, o por medio de dispositivos intermedios; transforma una magnitud física en un cambio eléctrico. Algunos sensores se pueden observar en la figura 10-1.

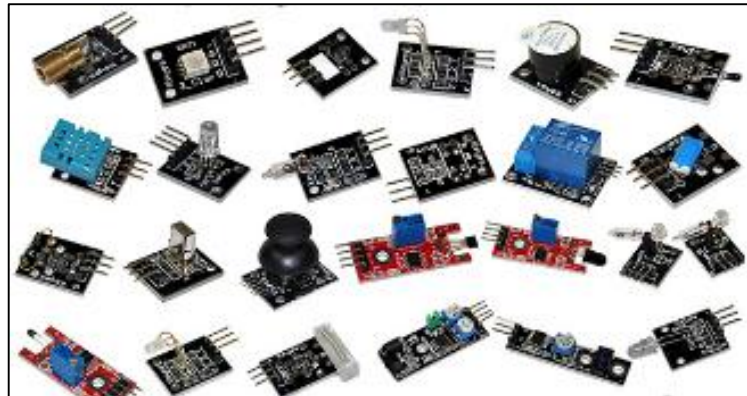


Figura 10-1: Sensores

Fuente: <https://shop.master.com.mx/product/detail?id=7554>

Las características y requerimientos que un sensor debe tener son un pequeño tamaño, un consumo bajo de energía, operar en densidades volumétricas altas, ser autónomo y funcionar desatendidamente y tener capacidad para adaptarse al ambiente.

Los sensores pueden estar clasificados en tres categorías como se menciona en (Cantillo, 2010, p.22):

- **Sensores pasivos omnidireccionales:** Los sensores pasivos captan los datos sin necesidad de manipular el entorno. Son autoalimentados y solo usan la energía para amplificar la señal analógica captada. No hay ninguna noción de ‘dirección’ involucrada en estas mediciones.
- **Sensores pasivos unidireccionales:** Son sensores pasivos que tienen bien definida la dirección desde donde deben captar la información. Un ejemplo típico es una cámara.

- **Sensores activos:** Este tipo de sensores sondan el ambiente, por ejemplo, un radar o un sonar o algún tipo de sensor sísmico que generan ondas expansivas a través de pequeñas explosiones.

1.9 Desarrollo de páginas web

Una página web se define como un documento electrónico el cual contiene información textual, visual y/o sonora que se encuentra alojado en un servidor y puede ser accesible mediante el uso de navegadores.

Una página web forma parte de una colección de otras páginas webs dando lugar al denominado sitio web el cual se encuentra identificado bajo el nombre de un dominio. (QUESS, 2017)

1.9.1 Lenguajes de programación

A medida que paso el tiempo, las tecnologías fueron desarrollándose y surgieron nuevos problemas para dar solución. Desarrollando lenguajes de programación para la web dinámicos, que permitieran interactuar con el personal encargado y utilizaran sistemas de Bases de Datos. Las páginas web se desarrollan en diferentes lenguajes para programación web como se observa en la tabla 5-1.

Tabla 5-1 Características de diferentes lenguajes de programación

Lenguaje de programación	Ventajas	Desventajas
HTML	<ul style="list-style-type: none"> • Sencillo que permite describir hipertexto. • Texto presentado de forma estructurada y agradable. • Lenguaje de fácil aprendizaje. • Lo admiten todos los exploradores. 	<ul style="list-style-type: none"> • Lenguaje estático. • La interpretación de cada navegador puede ser diferente. • Guarda muchas etiquetas que pueden convertirse en “basura” y dificultan la corrección. • El diseño es más lento. • Las etiquetas son muy limitadas.

JavaScript	<ul style="list-style-type: none"> • Lenguaje de scripting seguro y fiable. • Los scripts tienen capacidades limitadas, por razones de seguridad. • El código JavaScript se ejecuta en el cliente. 	<ul style="list-style-type: none"> • Código visible por cualquier usuario. • El código debe descargarse completamente.
PHP	<ul style="list-style-type: none"> • Se caracteriza por ser un lenguaje muy rápido. • Soporta en cierta medida la orientación a objeto. Clases y herencia. • Es un lenguaje multiplataforma: Linux, Windows, entre otros. • Capacidad de conexión de base de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, entre otras. • Es libre, por lo que se presenta como una alternativa de fácil acceso para todos. 	<ul style="list-style-type: none"> • Se necesita instalar un servidor web. • Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto, puede ser más ineficiente. • La programación orientada a objetos es aún muy deficiente para aplicaciones grandes. • Dificulta la organización por capas de la aplicación.
ASP	<ul style="list-style-type: none"> • Usa Visual Basic Script, siendo fácil para los usuarios. • Comunicación óptima con SQL Server. • Soporta el lenguaje JScript (JavaScript de Microsoft). 	<ul style="list-style-type: none"> • Código desorganizado. • Se necesita escribir mucho código para realizar funciones sencillas. • Tecnología propietaria. • Hospedaje de sitios web costosos.
ASP. NET	<ul style="list-style-type: none"> • Completamente orientado a objetos. • Facilita el mantenimiento de grandes aplicaciones. • Incremento de velocidad de respuesta del servidor. • Mayor velocidad. 	<ul style="list-style-type: none"> • Mayor consumo de recursos
JSP	<ul style="list-style-type: none"> • Ejecución rápida del servlets. • Crear páginas del lado del servidor. • Multiplataforma. • Código bien estructurado. • Integridad con los módulos de Java. • La parte dinámica está escrita en Java. • Permite la utilización de servlets. 	<ul style="list-style-type: none"> • Complejidad de aprendizaje.
Python	<ul style="list-style-type: none"> • Libre y fuente abierta. 	<ul style="list-style-type: none"> • Lentitud por ser un lenguaje interpretado.

	<ul style="list-style-type: none"> • Lenguaje de propósito general. • Gran cantidad de funciones y librerías. • Sencillo y rápido de programar. • Multiplataforma. • Licencia de código abierto (Opensource). • Orientado a Objetos. • Portable. 	
--	---	--

Fuente: <http://www.maestrosdelweb.com/los-diferentes-lenguajes-de-programacion-para-la-web/>
Realizado por : GUÑO Wilmer, 2017

1.9.2 Estructura del sitio web

Existen ciertos elementos de usabilidad con los que debe contar tu página web para permitir una fácil y rapidez encuentre cada elemento para lo cual existe determinadas características como:

1.9.2.1 Encabezado

Logotipo (no borroso, no mal recortado, no pixeleado). Bajo el logotipo, una frase corta que indique exactamente a qué se dedica tu empresa. Por ejemplo, bajo nuestro logotipo dice “Diseño y desarrollo de páginas y aplicaciones web”

1.9.2.2 Barra de navegación

Debe permanecer en el mismo lugar en todas las páginas interiores, aunque puede estar en otro lugar en la portada si hay una buena justificación. Al estar siempre en el mismo lugar el visitante sabe en todo momento donde buscar las demás opciones.

Los diferentes niveles de opciones deben indicarse visualmente con tipos de letra, negritas, colores o menús desplegables. Los textos de cada opción deben ser claros y concisos. (QUADRATO, 2017).

1.9.2.3 Área de contenido principal

Es el área donde la vista del visitante va después de ojear rápidamente el encabezado. Aquí debes poner tu información.

1.9.2.4 Barra lateral de contenido

En algunos casos es muy útil tener una barra lateral, en la cual se puede colocar información de fácil acceso, como números telefónicos o una liga a nuestro formulario de cotizaciones.

1.9.2.5 Pie de página

Debe contener un menú alternativo en puro texto, con las opciones de primer nivel. También úsalo para colocar información de contacto u otra información importante. El contenido del pie debe aparecer en todas las páginas.

1.10 Base de datos

Una base de datos (cuya abreviatura es BD) es una entidad en la cual se pueden almacenar datos de manera estructurada, con la menor redundancia posible. Diferentes programas y diferentes usuarios deben poder utilizar estos datos. Por lo tanto, el concepto de base de datos generalmente está relacionado con el de red, ya que se debe poder compartir esta información. Siguiendo una estructura jerárquica como se visualiza en la figura 11-1 (CCM, 2017)

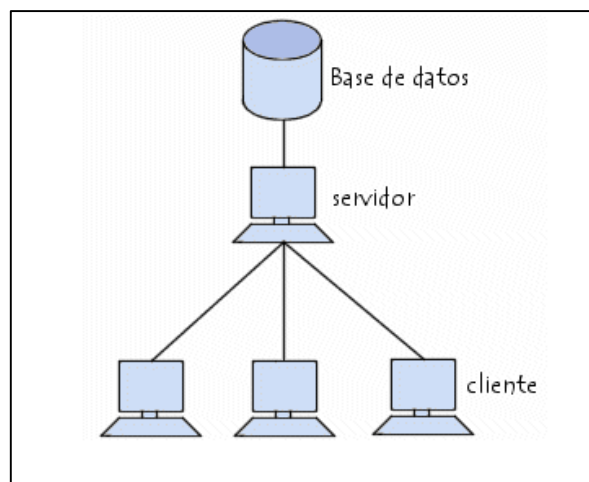


Figura 11-1: Estructura de una DB

Fuente: <http://es.ccm.net/contents/66-introduccion-a-las-bases-de-datos>

1.10.1 ¿Por qué se utiliza una base de datos?

Una base de datos proporciona a los usuarios el acceso a datos, que pueden visualizar, ingresar o actualizar, en concordancia con los derechos de acceso que se les hayan otorgado. Se convierte más útil a medida que la cantidad de datos almacenados crece.

Una base de datos puede ser local, es decir que puede utilizarla solo un usuario en un equipo, o puede ser distribuida, es decir que la información se almacena en equipos remotos y se puede acceder a ella a través de una red.

La principal ventaja de utilizar bases de datos es que múltiples usuarios pueden acceder a ellas al mismo tiempo. (CCM, 2017).

CAPITULO II

2. MARCO METODOLÓGICO

2.1 Introducción

En esta sección se especifica el diseño de hardware y software del prototipo realizado para el monitoreo de flujo de agua mediante GSM y radio frecuencia en un circuito local de agua potable. Se especifica los requerimientos técnicos que debe cumplir y se realiza el diseño en bloques de cada uno de sus módulos. Se justifica la selección de los sensores y componentes que lo integran. Al final se presenta el esquema de los módulos del sistema diseñado.

El alcance de la investigación pretende disponer de un prototipo que permita disminuir la fuga de agua en una malla de distribución, alertando mediante sms y vía web.

El método de recolección de información se lo realizó mediante la observación, mientras que la técnica de recolección de datos se lo realizó mediante toma de muestras de caudal de agua que pasa por la tubería para poder compararla y tomar los correctivos pertinentes.

.

2.2 Requerimientos del prototipo

En base al estudio realizado en el capítulo anterior se pueden definir los requerimientos de diseño que debe satisfacer el prototipo:

- Ser de bajo costo, de fácil instalación e interacción con el personal encargado.
- Las lecturas deben ser las correctas.
- Proporcionar información con comunicación inalámbrica RF en tiempo real.
- Emitir alertas de información con comunicación inalámbrica GSM al personal encargo.

- Almacenar en una base de datos local, periódicamente los valores obtenidos en la lectura del flujo de agua.
- Permitir el control del cierre/apertura de las electroválvulas.

2.3 Requerimientos de la arquitectura del prototipo

La estructura general propuesta para el prototipo se presenta en las siguientes figuras 1-2 y 2-2 donde se encuentran dos módulos fundamentales: Lectura, Transmisión y Activación; Recolección, Visualización y Control.

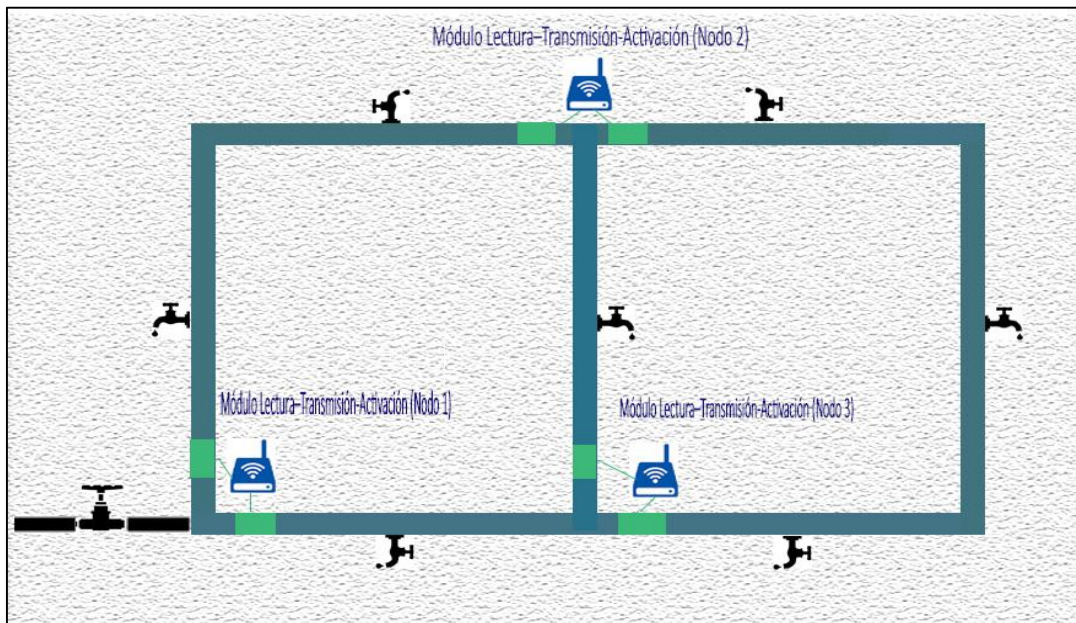


Figura 1-2: Concepción de arquitectura módulo Lectura, Transmisión y Activación
Realizado por : GUAÑO Wilmer, 2017

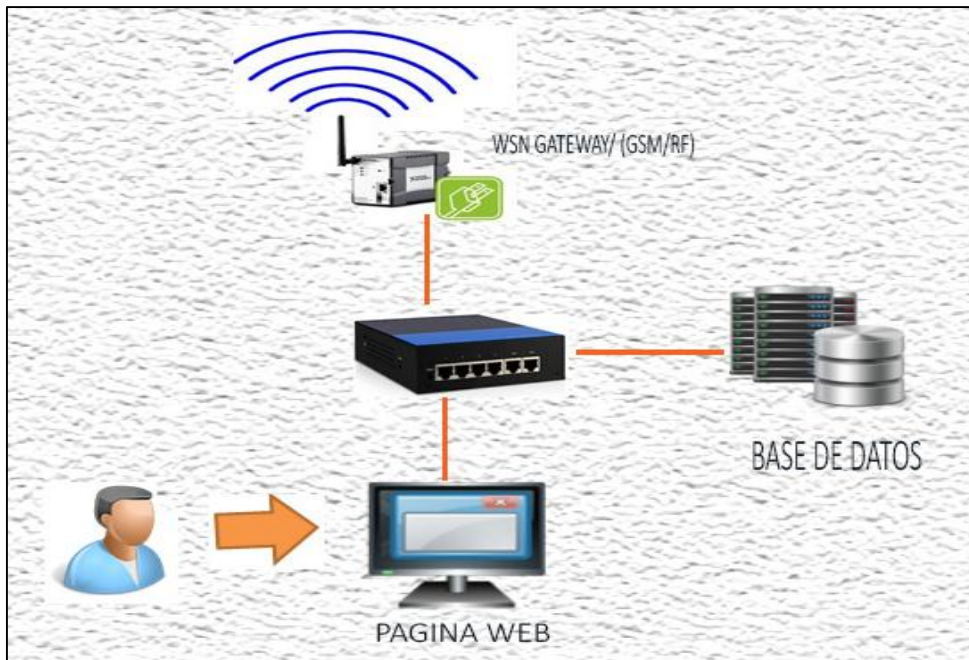


Figura 2-2: Concepción de arquitectura módulo Recolección, Visualización y Control
 Realizado por : GUAÑO Wilmer, 2017

2.3.1 Módulo de Lectura, Transmisión y Activación

Este módulo se encarga de enviar lecturas del sensor de flujo de caudal y recibir la información necesaria mediante RF para la activación de sus actuadores. Contiene una tarjeta de desarrollo Arduino que procesa la información y dispone de una fuente de alimentación.

2.3.2 Módulo de Recolección, Visualización y Control

Es el módulo que se encarga de asignar los valores de control para el cierre o apertura del paso del agua hacia cada ramal, comparando la información transmitida por los módulos de Lectura, Transmisión y Activación, esta información se procesa la cual determina si se encuentra dentro del porcentaje de uso de agua en el ramal de la red local.

Si se encuentra dentro del porcentaje establecido el prototipo emite un mensaje visual de aviso de habilitado en la página web que posee el prototipo previamente establecido, se encarga de emitir alertas mediante comunicación GSM al personal encargado cuando algún ramal posea falla en el abastecimiento del mismo.

La información obtenida y procesada se almacena en una base de datos local externa a la tarjeta de desarrollo, permitiendo acceder a reportes de la información que se encuentra en la base de datos.

2.4 Diseño de módulos del prototipo

Después de definir todos los requerimientos y la obtención del diseño de arquitectura del prototipo. Dichos diseños son realizados respectivamente para cada uno de los módulos.

2.4.1 Módulo de Lectura, Transmisión y Activación

Este módulo siempre inicializa con el bloque del sensor de flujo de agua, obtenido mediante sensores de flujo ubicados en la tubería, esta información es enviada al bloque de procesamiento de información en cual se le asigna a un vector la información de todos los resultados obtenidos por la tarjeta de desarrollo.

Enviando la información mediante el bloque de transmisor / receptor, este bloque también recibe información la cual es enviada al bloque de control, que se encarga del control de activación o desactivación este envía los datos necesarios al bloque de actuadores en cual se encuentran el circuito de potencia y las electroválvulas que permiten u obstruyen el paso del agua. Finalmente cuenta con un bloque alimentación de corriente continua.

En la figura 3-2 se observa el diagrama de bloques obtenido al desarrollar el módulo de Lectura, Transmisión y Activación donde se visualiza las diferentes interconexiones realizadas.

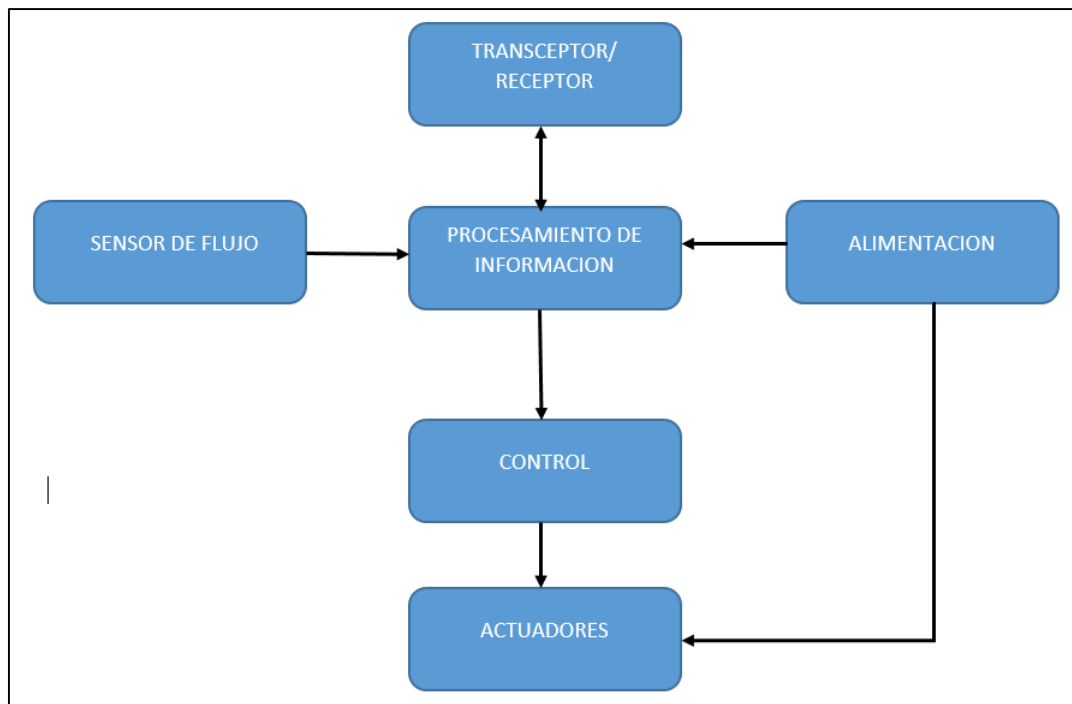


Figura 3-2: Arquitectura del módulo de Lectura, Transmisión y Activación
 Realizado por : GUAÑO Wilmer, 2017

2.4.2 Módulo de Recolección, Visualización y Control

El bloque de comunicación recibe la información del flujo de agua y el estado de los actuadores situados en el módulo Lectura, Transmisión y Activación, el bloque de procesamiento de información recibe estos datos, ordena y posteriormente envía al bloque de control el cual se encarga de ejecutar los controles respectivos sobre el estado de la red de agua potable que controla el prototipo.

Los datos después de ser tratados son enviados a los bloques de transmisor / receptor, alerta, visualización y de almacenamiento; el bloque de transmisor / receptor envía todos los controles que se debe ejecutar en el módulo Lectura, Transmisión y Activación.

El bloque de alerta se encarga de mandar un mensaje de texto al personal encargado del monitoreo, después de ser enviado el mensaje de alerta la información se registra en el bloque de almacenamiento el cual está conformado por una base de datos local permitiendo liberar espacio en nuestra tarjeta de desarrollo, la información antes de ser almacenada se visualiza mediante el bloque de resultados.

El bloque de resultados permite observar los datos obtenidos después de ser tratados y ordenados por el módulo Recolección, Visualización y Control permitiendo al personal encargado observar el funcionamiento de cada nodo que existe en el sistema, controlando y activando cuando este se desactiva al obtener problemas en el ramal que controla. Finalmente cuenta con el bloque de alimentación de corriente continua.

En la figura 4-2 se observa el diagrama de bloques obtenido al desarrollar el módulo de Recolección, Visualización y Control donde se visualiza las diferentes interconexiones realizadas.

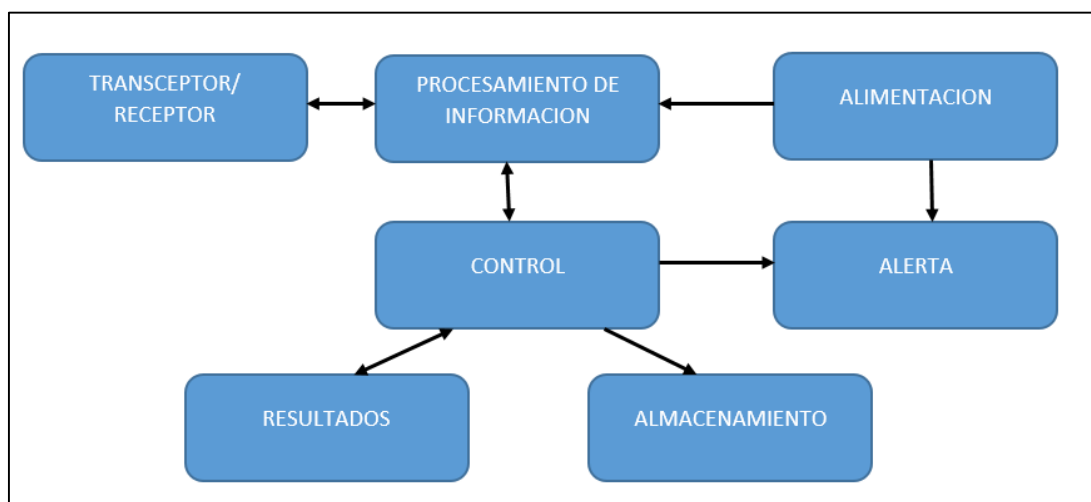


Figura 4-2: Arquitectura del módulo de Recolección, Visualización y Control
Realizado por : GUAÑO Wilmer, 2017

2.5 Selección de dispositivos para el desarrollo del prototipo

Cada uno de los dispositivos que se detallan a continuación conforman los módulos del prototipo de monitoreo de flujo de agua mediante GSM y radio frecuencia en un circuito local de agua potable y una breve reseña de su selección.

2.5.1 Tarjetas de desarrollo

El prototipo consta con tarjetas de desarrollo Arduino que cumple las características requeridas para la construcción del prototipo, esto se debe a su bajo costo y fácil adquisición en el mercado, número de terminales e interrupciones necesarias para los sensores de flujo que se utiliza, además posee un software libre y librerías que se actualizan constantemente para el desarrollo de nuevas aplicaciones.

Arduino presenta diferentes modelos que permiten escoger para su utilización, para poder escoger la tarjeta de desarrollo adecuada se realiza una tabla 1-2 para cada uno de los módulos según los requerimientos de cada uno de ellos.

Tabla 1-2: Modelos de Arduino seleccionados

Descripción	Mega	Nano
Precio en dólares	35	7
Microcontrolador	Atmega2560	Atmega328
Voltaje de entrada	7-12V	6-20V
Pines digitales Entrada/Salida	54	12
Entradas análogas	16	8
Memoria flash	256k	32KB
Memoria SRAM	8Kb	2 KB
Memoria EEPROM	4 KB	1 KB
Dimensiones	100 x 50 mm	18,5mm x 43,2mm
Comunicación SPI	2	1
Puertos seriales	4	1
Salida de alimentación a 3.3V	Si	Si
Interrupciones	6	2

Fuente: <http://saber.patagoniatec.com>

Realizado por : GUAÑO Wilmer, 2017

Para el módulo de Lectura, Transmisión y Activación se ha utilizado una tarjeta de desarrollo Arduino Nano que cumple los requisitos para utilización en este módulo, por su dimensión, número de interrupciones, alimentación de 3.3v, comunicación SPI y su bajo costo de adquisición en el mercado, permitiendo optimizar eficiencia y espacio en el prototipo; en la figura 5-2 se observa la placa de desarrollo Arduino Nano.



Figura 5-2: Arduino Nano

Fuente: http://img.dxcn.com/productimages/sku_81877_3.jpg

Para el módulo de Recolección, Visualización y Control se ha utilizado una tarjeta de desarrollo Arduino Mega que cumple los requisitos para utilización en este módulo, por su número de terminales analógicas y digitales, comunicación SPI, puertos seriales, gran espacio de memoria RAM y su gran compatibilidad con varios dispositivos o módulos externos a este, en la figura 6-2 se observa la placa de desarrollo Arduino Mega.

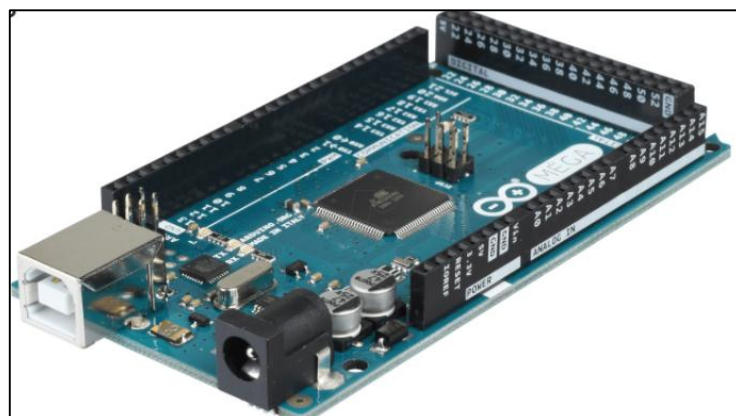


Figura 6-2: Arduino Mega

Fuente: <https://www.reichelt.nl/Single-board-microcontroller/ARDUINO-MEGA/>

2.5.2 Comunicación

Para la comunicación del prototipo se seleccionó tres dispositivos los cuales permiten la comunicación con la base de datos, envió y recepción de información entre los módulos, envió de alerta al personal encargado ante una perturbación en el circuito de agua potable cuando esta evita el paso de agua por su ramal que controla.

2.5.2.1 Módulo de comunicación nRF24L01

Se escogió este módulo de comunicación NRF24L01 por poseer comunicación SPI permitiendo un enlace más estable además se puede utilizarse al mismo tiempo un máximo de 6 canales lo que facilita enviar información desde los módulos de Lectura, Transmisión y Activación.

Esos datos son recibidos por el módulo de Recolección, Visualización y Control, las características de este módulo de comunicación inalámbrica de RF se observan en la tabla 2-2.

Tabla 2-2: Características del módulo Nrf24L01

Parámetros	Valor
Suministro mínimo de voltaje	1.9 v
Suministro mínimo de voltaje	3.6v
Máxima velocidad de datos	2000 kbps
Corriente de suministro en TX	11 mA
Corriente de suministro en RX	12.3 mA
Corriente de suministro	14 mA
Rango de temperatura	-40 a +85 °C
Sensibilidad en 1000kbps	-85 dBm
Distancia máxima sin antena	40 m

Fuente: https://www.sparkfun.com/datasheets/Components/nRF24L01_prelim_prod_spec_1_2.pdf
Realizado por : GUAÑO Wilmer, 2017

El módulo se puede observar en la figura 7-2, por sus características, bajo costo y accesibilidad en el mercado se escoge como módulo de comunicación RF entre los nodos establecidos y el punto base de recepción de datos.

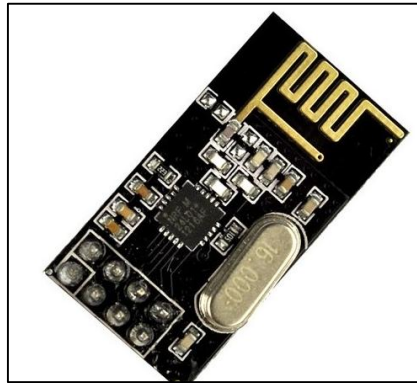


Figura 7-2: Módulo Nrf24L01

Fuente: <http://saber.patagoniatec.com/>

2.5.2.2 Módulo de comunicación Sim800L

Se escogió el módulo de comunicación Sim800L por su tamaño y bajo costo de adquisición; el módulo se encarga de enviar un mensaje de alerta al personal encargado, el módulo para su funcionamiento requiere una tarjeta SIM la cual permite anexas a la red de cobertura telefónica que se encuentre dentro de la zona de control, el módulo además posee una antena la cual ayuda a tener mejor cobertura, se encuentra ubicado en el módulo de Recolección, Visualización y Control. Dentro de sus características que posee el módulo SIM 800L podemos observar en la tabla 3-2:

Tabla 3-2: Características del módulo Sim800L

Descripción	Características
Quad-band	850/900/1800/1900mhz, se conectan a cualquier red mundial GSM (sim 2G)
Hacer y recibir	Llamadas de voz

Enviar y recibir	Mensajes SMS
Enviar y recibir datos	GPRS (TCP / IP, HTTP, etc.)
PWM / Buzzer	Control motor vibratorio
Dimensiones	27 x 39 mm

Fuente: <https://electronilab.co/tienda/modulo-celular-gsm-gprs-sim800l-v2-0-con-antena/>
Realizado por : GUANO Wilmer, 2017

El módulo con las anteriores especificaciones se puede observar en la figura 8-2 que se presenta continuación.



Figura 8-2: Módulo SIM800L

Fuente: <https://altronics.cl/mod-sim800l-evb>

2.5.2.3 Módulo de comunicación ethernet shield

El módulo shield ethernet se usa para la comunicación con la página web y la base de datos local externa, esta shield se escogió ya que podemos usar como servidor o cliente dependiendo los requerimientos que se utilice en el código principal que se ejecuta en nuestra tarjeta de desarrollo Arduino Mega, la shield ethernet se encuentra en el módulo de Recolección, Visualización y Control; se puede observar al módulo en la figura 9-2.



Figura 9-2: Módulo ethernet shield

Fuente: <https://electronilab.co/tienda/ethernet-shield-con-wiznet-w5100-internet-para-arduino/>

2.5.3 Sensor de flujo para el prototipo

El sensor de flujo YF-S201 fue escogido gracias a que posee varias aplicaciones que se realiza, por sus características que posee este sensor presentadas en la tabla 4-2, bajo costo en el mercado y su fácil adquisición del mismo satisfaciendo los requerimientos que necesita para el desarrollo del prototipo.

Tabla 4-2: Características del Sensor YF-S201

Descripción	Características
Modelo	YF-S201
Tipo de sensor	Efecto Hall
Voltaje de funcionamiento	5 a 18 V DC
Max consumo de corriente	15 mA a 5 V
Tipo de salida	5V TTL
Trabajo Caudal	de 1 a 30 litros / minuto
Temperatura de funcionamiento	-25 a 80 °C
Humedad de trabajo Rango	35% -80%
Precisión	± 2%
Rango de flujo	1-30L/min
Modo de detección	Vertical
Presión máxima del agua	2,0 MPa
Ciclo de trabajo de la salida	50% + 10%

Velocidad de flujo características del pulso	7,5 * Caudal (L / min)
Pulsos por litro	450
Durabilidad	300.000 ciclos
Tamaño	2.5 "x 1.4" x 1.4
conexiones nominales de tubería	1/2" diámetro

Fuente: <https://www.hetpro-store.com/TUTORIALES/sensor-de-flujo/>
Realizado por : GUANO Wilmer, 2017

El sensor YF-S201 está conformado por un cuerpo de plástico el cual recubre al mecanismo de funcionamiento que se encarga de generar los pulsos; en su interior se encuentra un rotor de agua y sensor de efecto hall, cuando el agua fluye a través del rotor esto producen cambios de velocidad en el mismo.

Su configuración para el funcionamiento está distribuida por tres cables de conexión: Vcc – Rojo, Gnd -Negro, Señal – Amarillo. El sensor YF-S201 se observa en la figura 10-2.



Figura 10-2: Sensor de flujo YF-S201

Fuente: <http://miarduinounotieneunblog.blogspot.com/2016/04/caudalimetro-con-sensor-de-flujo-yf.html>

Los pulsos obtenidos por el sensor de efecto hall son enviados a la tarjeta de desarrollo Arduino Nano para convertir en flujo volumétrico, según las especificaciones del fabricante el sensor de flujo debe ser calibrado con la ecuación 2-1.

$$k = \frac{n^k \text{Pulsos}}{\text{Volumen} * 60} \quad \text{Ecuación 2-1}$$

Donde:

K: Factor de conversión

n: Número de pulsos

V: Volumen de agua de prueba * 60 seg

2.5.4 Actuadores de control para el prototipo

Dentro de los actuadores de control se seleccionó una electroválvula de 12V dc normalmente cerrada la cual posee características que permiten su uso en el prototipo, por su bajo costo y fácil adquisición en el mercado ecuatoriano. En la tabla 5-2 se visualiza las características de la electroválvula en la figura 11-2.

Tabla 5-2: Características de la electroválvula 12V DC

Descripción	Característica
Voltaje de suministro	12v
Corriente de suministro	320mA
Presión de trabajo	0.02 Mpa – 0.8 Mpa
Temperatura de trabajo	1 °C – 75 °C
Tiempo de respuesta (open)	≤ 0.15 sec
Peso Tiempo de respuesta (close)	≤ 0.3 sec
Peso	3.4oz
Vida útil	50millones de ciclos
Dimensiones	3" x 2.25" x 2"

Fuente: <https://electronilab.co/tienda/electrovalvula-valvula-selenoide-agua-12v-dc-12/>
Realizado por : GUAÑO Wilmer, 2017



Figura 11-2: Electroválvula 12V DC

Fuente: <https://electronilab.co/tienda/electrovalvula-valvula-solenoid-agua-12v-dc-12/>

2.5.5 Alimentación del prototipo

Para el funcionamiento del prototipo se ha ubicado dos fuentes de alimentación, siendo un sistema de constante funcionamiento y para el cumplimiento de los requerimientos planteados.

Por esta razón el prototipo consta de una fuente alimentación para el circuito de censado, dentro de estos están la tarjeta de desarrollo, medios de comunicación y sensores, la fuente es una batería de litio 5v de 1200 mA conectada a un módulo tp 4056 el cual permite recargar baterías de litio mediante entrada micro USB o paneles solares, sus características se observa en la tabla 6-2.

Tabla 6-2: Especificaciones módulo Tp4056

Descripción	Características
Voltaje de entrada	5V
Voltaje de salida	4.2+/-1% V
Max. Corriente de salida	1.2A
Max. Potencia de salida	4.2W
Temperatura de trabajo	-20 ~ 85 °C
Eficiencia de conversión	80%
Dimensiones	1.1 in x 0.79 in x 0.2 in (2.8 cm x 2 cm x 0.5 cm)

Fuente: <http://www.dx.com/p/tp4056-4-2w-1-2a-5v-lithium-battery-charging-discharging-protection-module-w-micro-usb-blue>
Realizado por : GUAÑO Wilmer, 2017

Conectado al módulo tp4056 se encuentra un panel solar de 9v-100mA, el cual permite recarga la batería durante el día y posteriormente la batería solventa la energía cuando no exista la luz solar. Las figuras 12-2, la 13-2 y 14-2 muestra a la batería usada, el módulo para cargar baterías de litio y el panel solar respectivamente.



Figura 12-2: Batería de litio de 5v y 1200mA

Fuente: http://www.cetronic.es/sqlcommerce/ficheros/dk_93/productos/111051070-1.jpg

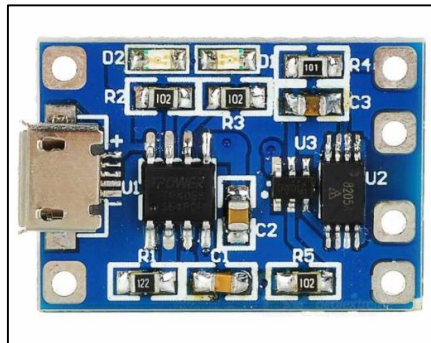


Figura 13-2: Módulo Tp4056

Fuente: http://img.dxcn.com/productimages/sku_397213_1.jpg

El panel solar es de capa fina la cual no está basada en silicio, sino de composición telurio de cadmio. Este tipo de panel solar se ocupa en aplicaciones tales como instalaciones domésticas y pequeños aparatos electrónicos, como cargadores, dispositivos con auto carga, etc.

Se escogió este tipo de placa solar por el costo ya que son más económicos en comparación a otro tipo de paneles solares y por la fácil adquisición en el mercado ecuatoriano.

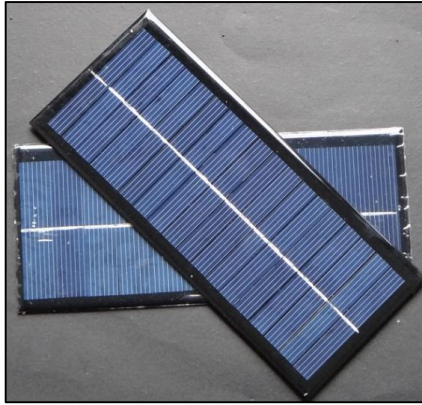


Figura 14-2: Panel solar 9v -100mA

Fuente:http://bigtronica.com/379-home_default/panel-solar-ensamble-5v-300ma.jpg

Debido a un mayor consumo de voltaje y corriente se ha seleccionado una fuente que permitirá suministrar la suficiente corriente y voltaje a la electroválvula y tener un correcto funcionamiento de la misma. La fuente se observa en la figura 15-2.



Figura 15-2: Fuente de 900W

Fuente:<http://www.arreglamipc.com/wp-content/uploads/2010/10/voltaje-de-una-fuente-de-poder.jpg>

2.6 Conexiones del prototipo

Después de seleccionar los dispositivos que cumplen los requerimientos para el desarrollo de cada módulo del prototipo de monitoreo de flujo de agua mediante GSM y radio frecuencia en un circuito local de agua potable, se realizaron en Fritzing las simulaciones respectivas de cada módulo del prototipo.

Para el desarrollo de las respectivas simulaciones se instaló las librerías que permiten la compatibilidad con Arduino Mega y Nano, posteriormente se simuló los respectivos circuitos para la comprobación del funcionamiento y evitar problemas al momento de su implementación.

2.6.1 Esquema de conexión del módulo Lectura, Transmisión y Activación

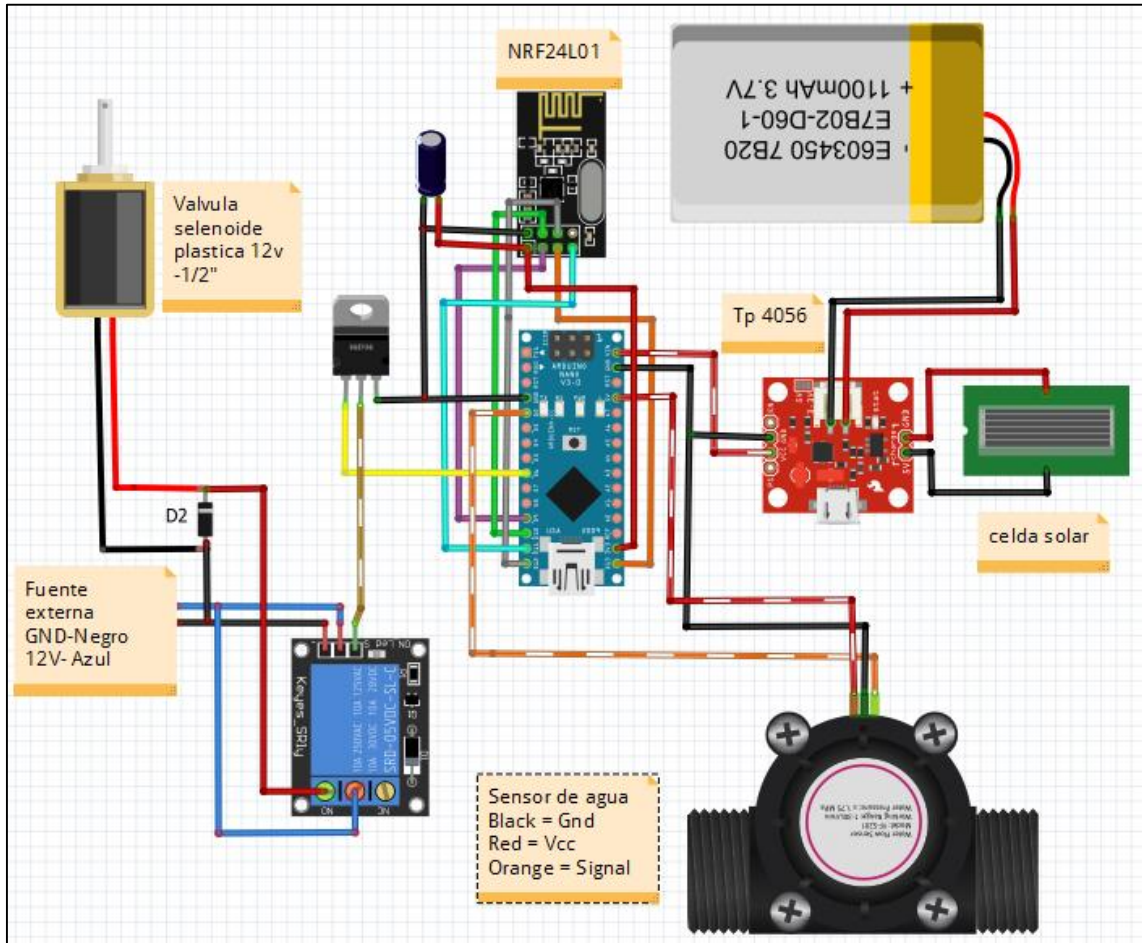


Figura 16-2: Esquema conexión del módulo Lectura, Transmisión y Activación

Realizado por : GUAÑO Wilmer, 2017

Esquema de conexión del módulo Lectura, Transmisión y Activación desarrollado en Fritzing obteniendo resultados como se observa en la figura 16-2, el componente principal es el Arduino Nano que se conecta con cada uno de los dispositivos que se necesita:

- La obtención de la señal del sensor de flujo YF-S201 se conecto en la terminal D2 y D3 del Arduino Nano por ser entradas que poseen interrupciones.
- Para la conexión entre Arduino Nano con el módulo NRF24L01, se utiliza la siguiente tabla 7-2.

Tabla 7-2: Pines de conexión Arduino Nano y NRF24L01

Pin	NRF24L01	Arduino Nano
GND	1	GND
VCC	2	3.3
CE	3	9
CSN	4	10
SCK	5	13
MOSI	6	11
MISO	7	12

Fuente: <http://www.prometec.net/nrf2401/#>
Realizado por : GUAÑO Wilmer, 2017

- En los terminales D6 y D7 se conecta con un transistor TIP121 este realiza la función de switch para la activación de los relés.
- Cada relé está conectado a una electroválvula de 12 v Dc que permite la apertura o cierre de agua por la tubería que se encuentran controlando.
- Los pines de Vin Y GND del Arduino Nano se encuentran conectados a los pines OUT+ y OUT- del módulo Tp4056 respectivamente para su alimentación.
- En el módulo Tp 4056 se encuentran conectados una batería de litio de 3.7v -1200mA en los pines B+ y B- del módulo, el panel solar se encuentra conectado a los pines + y – que se encuentran en el módulo.

2.6.2 Esquema de conexión del módulo Recolección, Visualización y Control

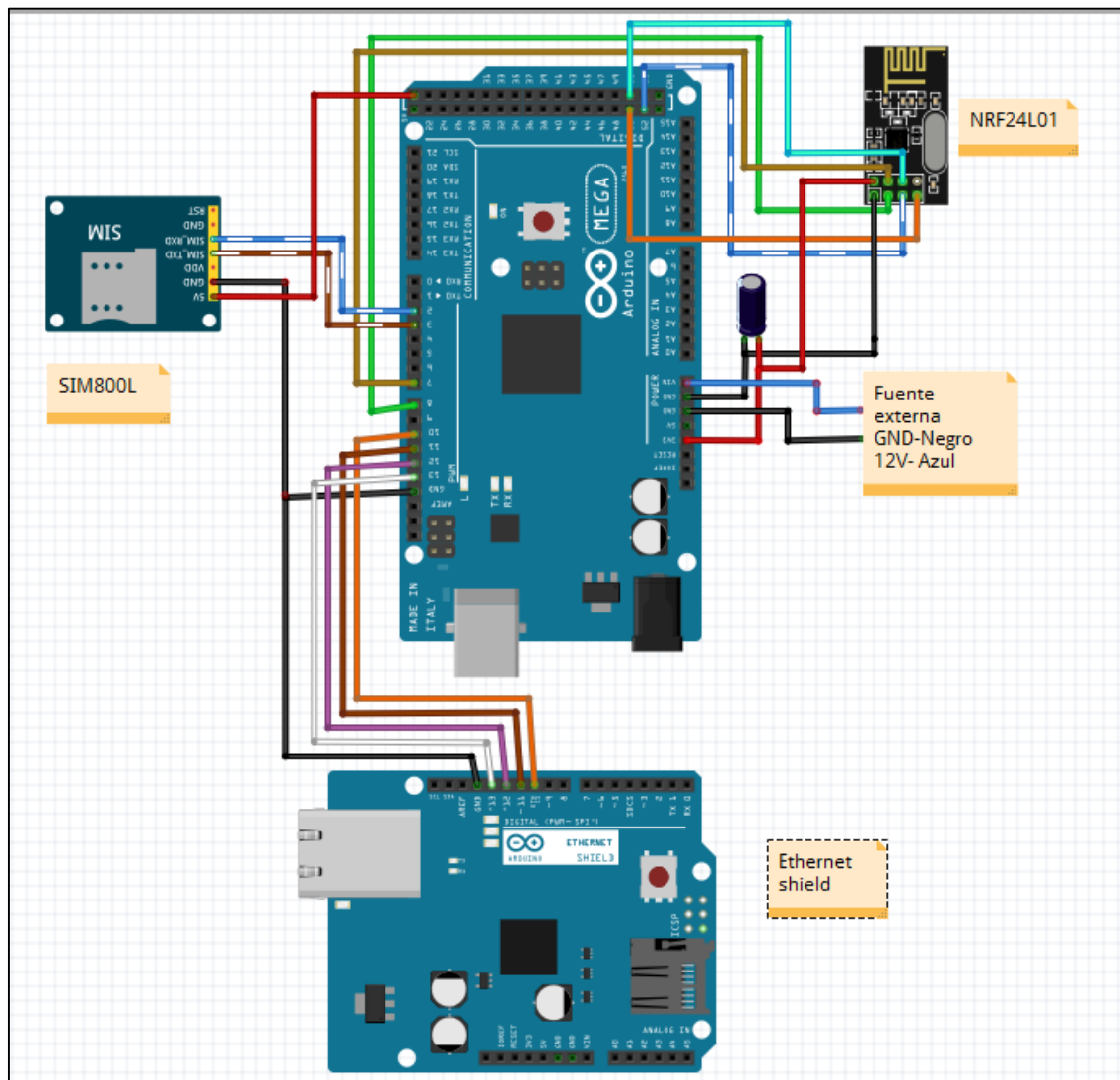


Figura 17-2: Esquema conexión del módulo Recolección, Visualización y Control
Realizado por : GUANO Wilmer, 2017

Esquema de conexión del módulo Recolección, Visualización y Control desarrollado en Fritzing obteniendo resultados como se observa en la figura 17-2, los componentes principales son el Arduino Mega y la Ethernet shield estos se conectan con cada uno de los dispositivos:

- Este módulo recibe los datos enviados por el módulo Lectura, Transmisión y Activación los datos se obtiene mediante el dispositivo NRF24L01, sus conexiones se visualizan en la tabla 8-2.

Tabla 8-2: Pines de conexión Arduino Mega y NRF24L01

Pin	NRF24L01	Arduino Mega
GND	1	GND
VCC	2	3.3
CE	3	8
CSN	4	7
SCK	5	52
MOSI	6	51
MISO	7	50

Fuente: <http://www.prometec.net/nrf2401/#>

Realizado por : GUANO Wilmer, 2017

- La tarjeta de desarrollo Arduino Mega además está conectado con el módulo de comunicación Ethernet shield este permite la comunicación con el router mediante el puerto RJ45. Permitiendo la visualización y envío de la información a la base de datos local externa.
- El módulo de comunicación SIM800L permite el envío de una señal de alerta al personal encargado a su teléfono móvil. Las conexiones que necesita el módulo de comunicación Sim800L con la tarjeta de desarrollo para su funcionamiento están visualizadas en la tabla 9-2.

Tabla 9-2: Pines de conexión Arduino Mega y SIM 800L

Pin SIM800L	Arduino Mega
GND	GND
5VIN	5V
TXD	2
RXD	3

Fuente: <http://i2.wp.com/albsquared.com/wp-content/uploads/2015/08/SIM800L.png>

Realizado por : GUANO Wilmer, 2017

2.7 Desarrollo del código del prototipo

El desarrollo de código para el funcionamiento del Arduino Nano y Mega fue realizado en el entorno integrado IDE de Arduino el cual está disponible para varios sistemas operativos como Windows, Mac y Ubuntu por ser un entorno de desarrollo estudiantil. Para trabajar se selecciona el modelo de Arduino que se utiliza, configurando la comunicación entre el entorno de desarrollo IDE que se encuentra instalado en la PC y el puerto conectado a la tarjeta de desarrollo Arduino.

2.7.1 *Requerimientos del prototipo*

- Determinar el flujo de agua asignado
- Tiempo de ejecución del programa rápido
- Transmisión de datos en tiempo real
- Interfaz amigable
- Control para el cierre y apertura de la electroválvula
- Formato de almacenamiento debe ser reconocido por software.

2.7.2 *Diagrama de flujo del programa del módulo Lectura, Transmisión y Activación*

En el diagrama de flujo de la figura 18-2 detalla la secuencia del programa desarrollado en IDE de Arduino.

Ciclo de inicialización

- Declaración de las todas las librerías usadas.
- Declaración de las variables globales.

- Declaración de las direcciones de los canales ocupados para comunicación entre los módulos
- Declaración de las variables utilizadas para los pines de los sensores y de los transistores.
- Configuración de entradas, salidas e interrupciones del Arduino Nano.
- Inicialización de la comunicación SPI del módulo NRF24L01.
- Inicialización del puerto Serial

Ciclo de procesamiento de datos

- Declaración de las funciones **ContarPulsos()** y **ObtenerFrecuencia ()**.
- La función **ContarPulsos()** se activa con las interrupciones que se genera con los datos obtenidos por el sensor.
- La función **ObtenerFrecuencia ()** almacena en variables globales los datos obtenidos por la función **ContarPulsos()**.
- La función **void loop ()** se encarga de inicializar cada función cuando sea requerida; se realiza los cálculos necesarios para obtención del flujo de agua que es tomado por el sensor.
- Se almacena en una variable vector llamada **datos**, esta información se envía por radio frecuencia hacia el módulo de Recolección, Visualización y Control.
- Se cierra el canal de comunicación que permite escribir y se inicializa el canal de escuchar, el cual posee una variable vector denominada **estado**.
- La variable **estado** almacena los datos de control enviados por el módulo Recolección, Visualización y Control.
- Con los datos de control se activa los pines de salida y se finaliza la función void loop ().

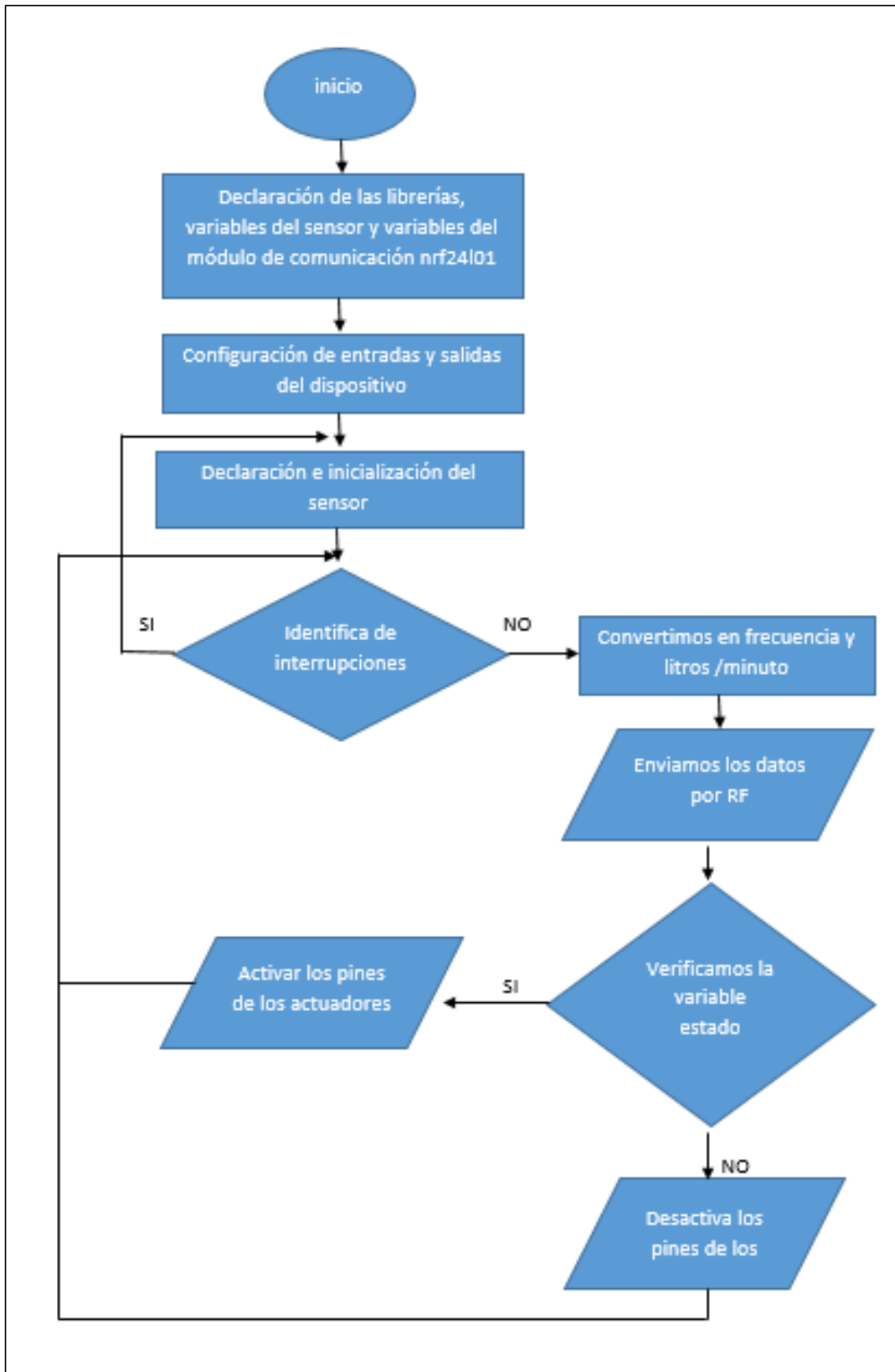


Figura 18-2: Diagrama de flujo del módulo Lectura, Transmisión y Activación
 Realizado por : GUAÑO Wilmer, 2017

2.7.3 *Diagrama de flujo del programa del módulo Recolección, Visualización y Control*

En el diagrama de flujo de las figuras 19-2 y 20-2 detalla la secuencia del programa desarrollado en IDE de Arduino.

Ciclo de inicialización

- Declaración de las todas las librerías usadas.
- Declaración de las variables globales.
- Declaración de las direcciones de los canales ocupados para comunicación entre los módulos
- Declaración de las variables para configuración de la dirección IP y MAC utilizadas por el módulo de comunicación ethernet shield
- Configuración de los pines utilización para el módulo de comunicación SIM800L
- Inicialización de la comunicación SPI del módulo NRF24L01.
- Inicialización del módulo e comunicación SIM800L.
- Inicialización del módulo e comunicación ethernet shield.
- Inicialización del puerto Serial

Ciclo de procesamiento de datos

- Declaración de las funciones **CONTROLdeRED ()**, **nodes ()**, **promedios ()**, **CONSM ()** y **servidorr ()**.
- La función **promedios ()** promedia toda la información durante un tiempo de muestro establecido de 1.20s obtenida mediante los módulos de comunicación, la información es previamente almacenados ordenadamente en vectores variables,
- La función **CONTROLdeRED ()** se encarga de procesar todos los datos obtenidos previamente en la función de promedios, controlando cada red que se encuentre con problemas de fallas durante su monitoreo y evitar desperdicios de agua.
- La función **nodes ()** permite la visualización de las alertas en la página web diseñada.

- La función **servidorr()** envía la información y conecta con lavase de datos local externa.
- La función **CONSM ()** envía el mensaje de alerta mediante comunicación GSM al personal encargado encarga de la monitorización.

Ciclo de repetición

- La función **void loop ()** se encarga de inicializar cada función cuando sea requerida; se realiza los cálculos necesarios.
- Se inicializa verificando la conexión del shield Ethernet para poder visualizar y obtener datos de activación.
- Dentro de esta función esta la página web desarrollada para poder visualizar y controlar mediante ella
- Se cierra el canal de comunicación que permite escuchar y se inicializa el canal de escribir el cual posee una variable vector denominada **estado** que es enviado a los módulos *Lectura, Transmisión y Activación*.
- Con los datos de control enviados y se finaliza la función void loop ().

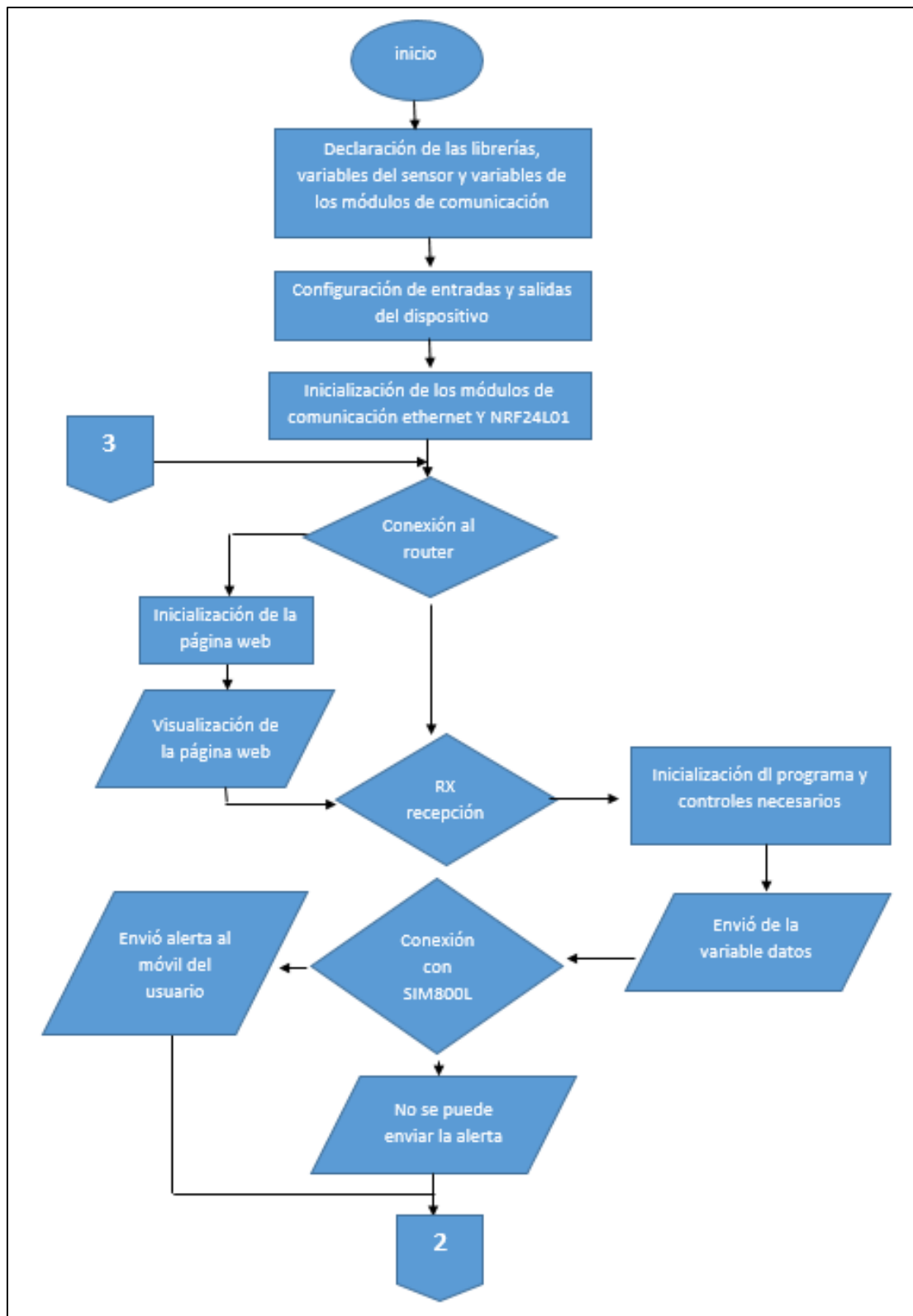


Figura 19-2: Diagrama de flujo del módulo Recolección, Visualización y Control .1
 Realizado por : GUAÑO Wilmer, 2017

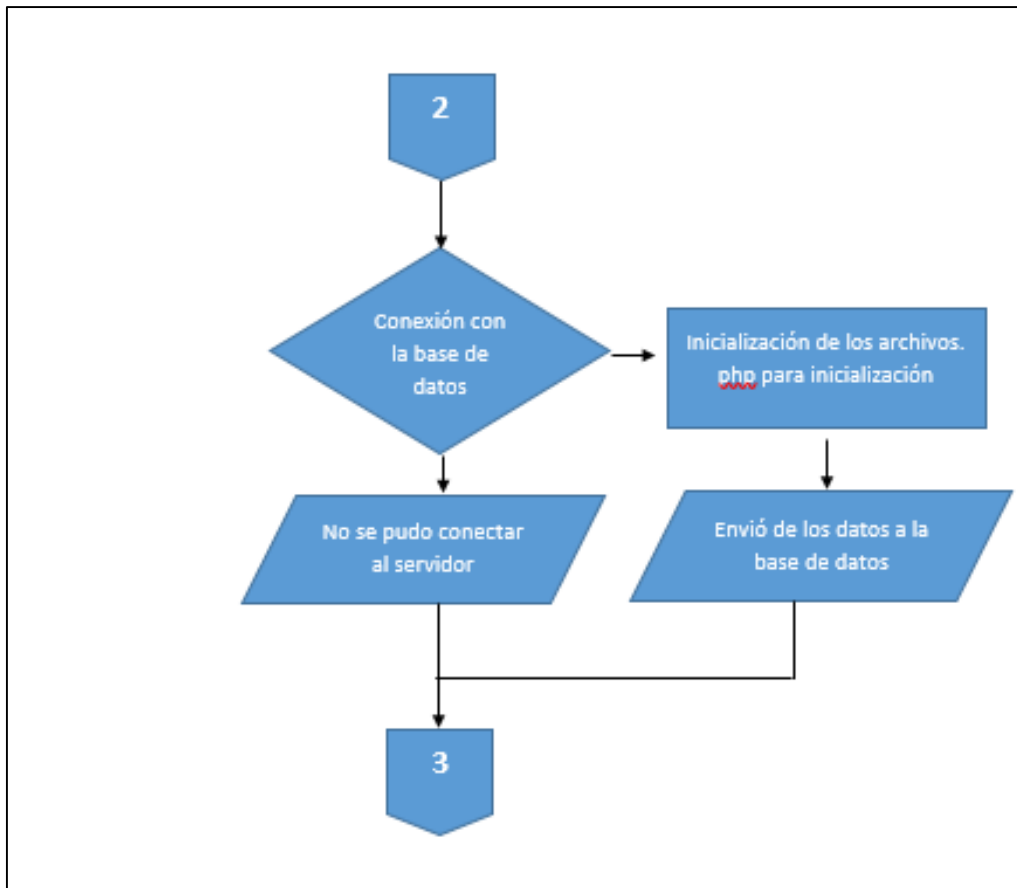


Figura 20-2: Diagrama de flujo del módulo Recolección, Visualización y Control .2
Realizado por : GUAÑO Wilmer, 2017

CAPITULO III

3. PRUEBAS Y RESULTADOS

3.1 Introducción

En este capítulo se presentan los resultados obtenidos en las pruebas realizadas con cada uno de los módulos que conforman el prototipo de monitoreo de flujo de agua mediante GSM y radio frecuencia en un circuito local de agua potable tanto como en el hardware y el software, se caracterizó la respuesta de cada módulo en función a los requerimientos del proyecto presentados en el capítulo anterior, para ello se realizó pruebas de funcionamiento que corroboren las variables medidas y sus errores de funcionamiento. Adicionalmente se presenta un análisis de costos.

3.2 Caracterización del prototipo

Para determinar que el prototipo cumpla con los requerimientos establecidos se realizaron pruebas con las variables medidas, variables de control y distintas formas de comunicación que se realiza.

Al no disponer de dispositivos costosos para mediciones comunicación y control con mejor exactitud que los dispositivos usados en el prototipo desarrollado, se decidió comprar dispositivos comerciales con características similares. Además, que los dispositivos para censar implementados en el prototipo tienen salidas digitales y tiene un error mínimo de calibración otorgados por la fábrica que los elabora este criterio se aplica a las características de los sensores usados para el flujo de agua.

3.3 Caracterización del módulo Recolección, Visualización y Control

En la figura 1-3 se observa una fotografía del prototipo del módulo de Recolección, Visualización y Control. En la parte superior e inferior se encuentran los distintos módulos de comunicación inalámbrica, módulo de comunicación Ethernet el cual está conectado con un cable Ethernet y la tarjeta de desarrollo Arduino Mega.

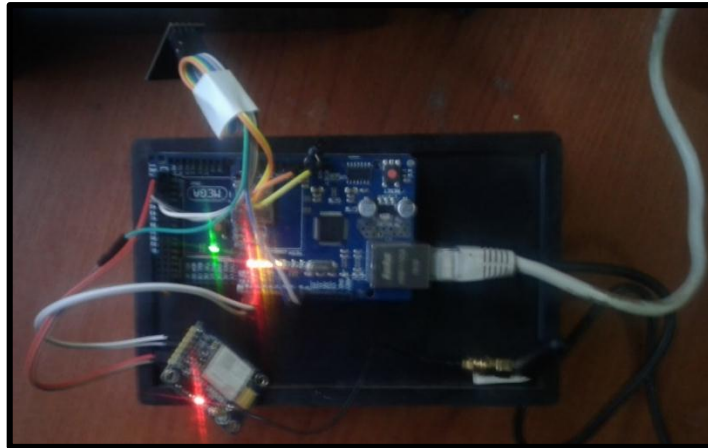


Figura 1-3: Fotografía del módulo Recolección, Visualización y Control
Realizado por : GUAÑO Wilmer, 2017

Para su funcionamiento se encuentran alimentados por una fuente externa de computadora la cual proporciona un voltaje y amperaje óptimo.

3.4 Caracterización del módulo Lectura, Transmisión y Activación

En las figuras 2-3, 3-3, 4-3 se visualizan las fotografías de los nodos 1,2,3 respectivamente los cuales conforman el módulo Lectura, Transmisión y Activación. Se programó las placas de desarrollo de forma que permitan introducir un número superior a los 6 canales que activan el módulo NRF24L-01 al mismo tiempo.

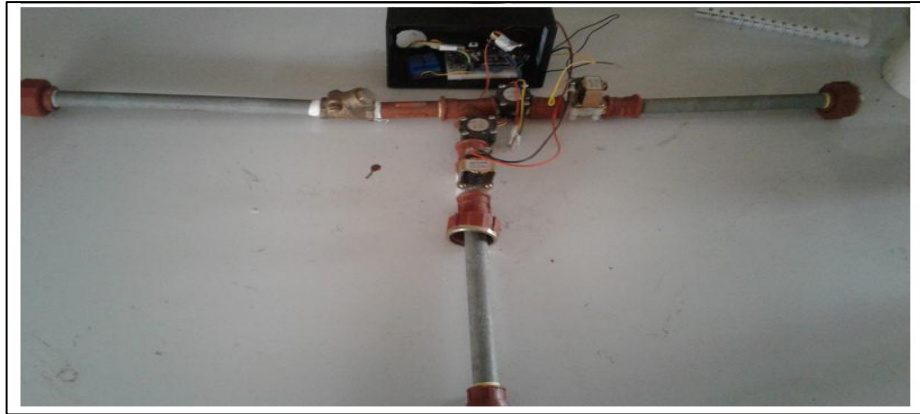


Figura 2-3: Fotografía del módulo Lectura, Transmisión y Activación (NODO 1)
Realizado por : GUAÑO Wilmer, 2017

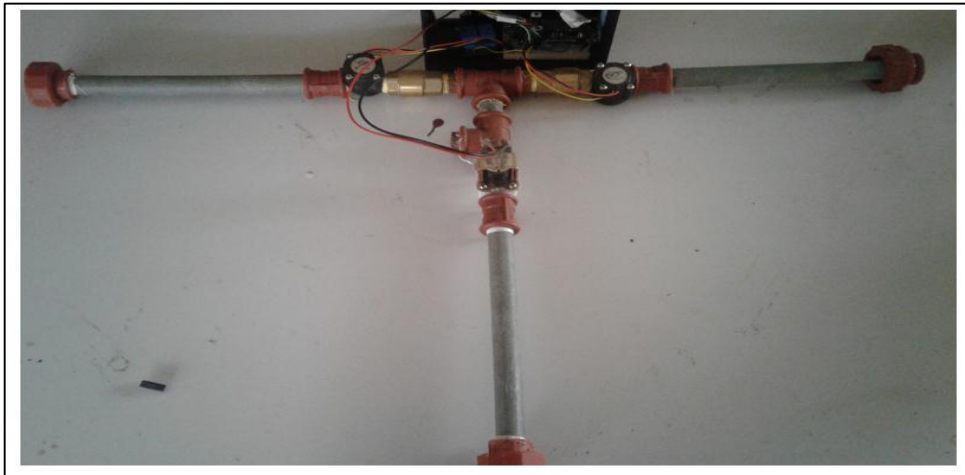


Figura 3-3: Fotografía del módulo Lectura, Transmisión y Activación (NODO 2)
Realizado por : GUAÑO Wilmer, 2017

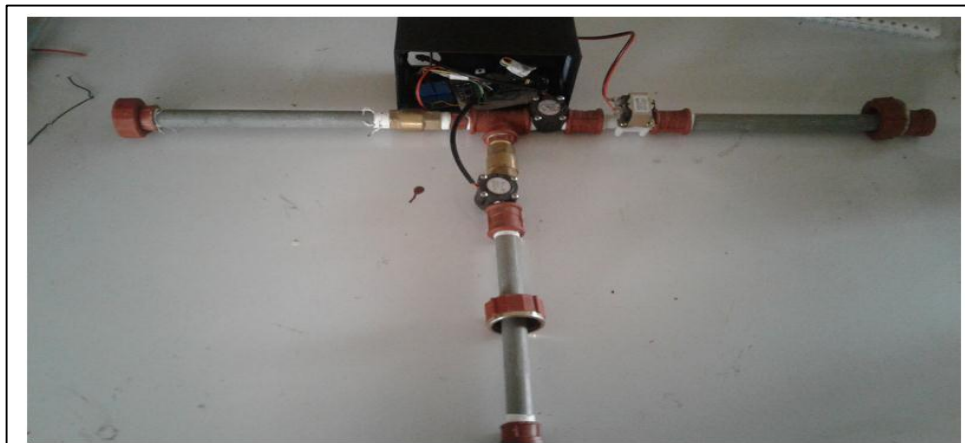


Figura 4-3: Fotografía del módulo Lectura, Transmisión y Activación (NODO 3)
Realizado por : GUAÑO Wilmer, 2017

En la parte superior de cada una de las figuras se puede observar la conexión de los sensores de flujo a la tubería de la red de agua, a su vez se observa las electroválvulas. Los dos dispositivos se encuentran conectados a la tarjeta de desarrollo Arduino Nano. Las figuras 2-3 ,3-3, 4-3 poseen características diferentes para satisfacer las necesidades de cada posición ubicada dentro de la red local de agua potable.

Cada nodo permite tomar datos de flujo de agua y ser enviados por medio del módulo de comunicación inalámbrica RF al módulo Recolección, Visualización y Control.

3.5 Caracterización de la red local de agua

Se realizó la implementación de una red local de agua potable para la implantación del prototipo, el cual posee una dimensión de 4m de largo y 3m de ancho en total dividida en dos mallas por medio de una tubería intermedia, como se observa en la figura 5-3. El circuito local de agua potable fue desarrollado con características de un circuito cerrado o mallado



Figura 5-3: Fotografía de la red local de agua potable
Realizado por : GUAÑO Wilmer, 2017

Posee 6 válvulas las cuales permiten la simulación de abastecimiento de agua a los usuarios y 6 válvulas extras que permiten la simulación de fallas de agua en una red local de agua potable.

3.6 Caracterización del flujo de agua

Se realizaron pruebas para la comprobación del funcionamiento de los sensores ocupados que son detallados a continuación.

3.6.1 Repetitividad del Lectura, Transmisión y Activación

Para la obtención de los datos se tomó como referencia a 8 litros de agua realizando 15 repeticiones por cada sensor implementado en el módulo para la obtención del porcentaje de error de cada uno. Esto se realizó por recomendaciones del fabricante. Los datos obtenidos se almacenaron en una hoja de cálculo Excel, obteniendo un análisis estadístico del error.

Los datos fueron tomados con una constante $k=7.5$ que es el factor de conversión que posee el fabricante para la obtención del caudal, estos datos obtenidos se muestran en las tablas 1-3 y 2-3.

Tabla 1-3: Prueba de repetitividad sensores: 1, 2,3

N ^a	Sensor 1		Sensor 2		Sensor 3	
	Volumen censado-litros	Error	Volumen censado-litros	Error	Volumen censado-litros	Error
1	9	1	9	1	8	0
2	10	2	9	1	8	0
3	8	0	10	2	10	2
4	8	0	8	0	9	1
5	9	1	8	0	10	2
6	9	1	9	1	9	1
7	10	2	8	0	9	1
8	8	0	10	2	8	0
9	9	1	9	1	8	0
10	10	2	10	2	9	1

11	10	2	8	0	8	0
12	9	1	9	1	8	0
13	8	0	10	2	9	1
14	10	2	10	2	9	1
15	9	1	8	0	8	0

Realizado por : GUAÑO Wilmer, 2017

Tabla 2-3: Prueba de repetitividad sensores: 4, 5,6

N ^a	Sensor 4		Sensor 5		Sensor 6	
	Volumen censado-litros	Error	Volumen censado-litros	Error	Volumen censado-litros	Error
1	8	0	8	0	10	2
2	8	0	8	0	8	0
3	10	2	10	2	9	1
4	10	2	9	1	10	2
5	9	1	9	1	10	2
6	9	1	8	0	8	0
7	8	0	8	0	9	1
8	9	1	8	0	9	1
9	10	2	10	2	8	0
10	9	1	8	0	10	2
11	8	0	9	1	8	0
12	9	1	10	2	8	0
13	8	0	8	0	9	1
14	9	1	10	2	10	2
15	10	2	10	2	8	0

Realizado por : GUAÑO Wilmer, 2017

El análisis estadístico se realizó con ayuda de Minitab con el que se calculó la media, la desviación estándar y el coeficiente de variación. El resultado de las mediciones de los sensores se presenta en la tabla 3.3.

Tabla 3-3: Estadística descriptiva para la repetitividad de monitoreo

Estadística descriptiva						
Resultados obtenidos	Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5	Sensor 6
Media	1,067	1	0,667	0,933	0,867	0,933
Error típico	0,206	0,218	0,187	0,206	0,236	0,228
Mediana	1	1	1	1	1	1
Moda	1	0-1-2	0	1	0	0
Desviación estándar	0,799	0,845	0,724	0,799	0,915	0,884
Varianza de la muestra	0,638	0,714	0,524	0,638	0,838	0,781
Curtosis	-1,35	-1,62	-0,65	-1,35	-1,86	-1,78
Coeficiente de asimetría	-0,13	0	0,63	0,13	0,29	0,14
Rango	2	2	2	2	2	2
Mínimo	0	0	0	0	0	0
Suma	16	15	10	14	13	14
Cuenta	15	15	15	15	15	15

Realizado por : GUAÑO Wilmer, 2017

3.6.2 Calibración del sensor de flujo

Debido a que las mediciones de los diferentes sensores presentan un error alto y por recomendación del fabricante se realiza la calibración de cada sensor de flujo de agua, disminuyendo su margen de error. Para lo cual se realizó pruebas para determinar el número de pulsos obtenidos en un determinado volumen de agua.

Mediante la tarjeta de desarrollo Arduino Nano se procede a cargar el programa de calibración ubicado en el anexo I, el proceso que realiza es emitir un número determinado de pulsos

dependiendo la cantidad de volumen de agua, en la tabla 4-3 se observa los datos obtenidos para cada sensor de flujo.

Tabla 4-3: Calibración de los sensores en función de pulsos

N ^a	Volumen (Litros)	Pulsos Sensor 1	Pulsos Sensor 2	Pulsos Sensor 3	Pulsos Sensor 4	Pulsos Sensor 5	Pulsos Sensor 6
1	4	1753	1712	1758	1656	1602	1672
2	3	1342	1353	1328	1264	1242	1319
3	2	926	898	923	850	820	814
4	4	1740	1742	1758	1602	1560	1663
5	2	908	846	923	882	852	855
6	6	2493	2484	2488	2390	2380	2424
7	3	1384	1354	1411	1279	1278	1202
8	5	2092	2054	2028	2054	2056	2070
9	2	912	894	941	880	865	879
10	6	2481	2452	2444	2378	2411	2415
Promedio de pulsos		1603,1	1578,9	1600,2	1523,5	1506,6	1526,9
Factor de conversión k		7,34	7,20	7,36	6,96	6,85	6,93

Realizado por : GUAÑO Wilmer, 2017

Obtenido el promedio de cada uno de los sensores de sus pulsos respectivos. Se determinó el factor de conversión de cada sensor que se calcula con la ecuación 1-2.

3.6.3 Comprobación de la calibración del Sensor de flujo

Después de haber obtenido para cada sensor su propio factor de conversión, se procedió a calibrar cada sensor de flujo YF-S201. Se realiza pruebas para determinar el margen de error que se obtiene después de calibrarse. El proceso de verificación se realizó tomando 12 muestras del volumen de agua aleatoriamente por cada sensor.

Tabla 5-3: Pruebas del sensor 1, 2 ,3 después de la calibración

N ^a	Volumen Referencia -litros	Sensor 1		Sensor 2		Sensor 3	
		Volumen censado-litros	Error Absoluto %	Volumen censado-litros	Error Absoluto %	Volumen censado-litros	Error Absoluto %
1	4	4,142	-0,142	3,813	0,187	4,07	-0,07
2	3	3,26	-0,26	2,954	0,046	3,121	-0,121
3	2	1,897	0,103	1,863	0,137	2,176	-0,176
4	4	4,062	-0,062	4,072	-0,072	3,921	0,079
5	2	1,922	0,078	1,831	0,169	2,157	-0,157
6	6	6,087	-0,087	5,859	0,141	6,01	-0,01
7	3	3,156	-0,156	2,896	0,104	2,991	0,009
8	5	4,941	0,059	5,036	-0,036	5,066	-0,066
9	2	1,866	0,134	2,109	-0,109	2,185	-0,185
10	6	6,351	-0,351	5,875	0,125	6,112	-0,112
11	7	7,128	-0,128	7,034	-0,034	7,104	-0,104
12	8	8,152	-0,152	7,882	0,118	8,099	-0,099

Realizado por : GUAÑO Wilmer, 2017

Tabla 6-3: Pruebas del sensor 4, 5 ,6 después de la calibración

N ^a	Volumen Referencia -litros	Sensor 4		Sensor 5		Sensor 6	
		Volumen censado-litros	Error Absoluto %	Volumen censado-litros	Error Absoluto %	Volumen censado-litros	Error Absoluto %
1	4	3,897	0,103	4,149	-0,149	3,914	0,086
2	3	2,931	0,069	3,026	-0,026	2,95	0,05
3	2	1,859	0,141	2,1	-0,1	2,016	-0,016
4	4	4,016	-0,016	3,946	0,054	4,091	-0,091

5	2	2,09	-0,09	1,87	0,13	1,93	0,07
6	6	6,026	-0,026	6,09	-0,09	5,944	0,056
7	3	2,875	0,125	3,138	-0,138	2,902	0,098
8	5	4,879	0,121	5,198	-0,198	5,048	-0,048
9	2	2,002	-0,002	1,98	0,02	1,954	0,046
10	6	5,891	0,109	6,187	-0,187	5,875	0,125
11	7	6,98	0,02	6,932	0,068	6,802	0,198
12	8	8,036	-0,036	8,136	-0,136	8,006	-0,006

Realizado por : GUAÑO Wilmer, 2017

En las tablas 5-3 y 6-3 se muestra el volumen de agua de referencia versus el volumen de agua medido después que los sensores fueron calibrados, el resultado del análisis estadístico experimental realizado permite verificar el porcentaje de error menor al anterior antes de ser calibrados los sensores. Este resultado se puede observar en la tabla 7-3.

Tabla 7-3: Estadística descriptiva del error después de la calibración

Estadística descriptiva						
Resultados obtenidos	Sensor 1	Sensor 2	Sensor 3	Sensor 4	Sensor 5	Sensor 6
Media	-0,0803	0,0647	-0,0843	0,0432	-0,0627	0,0473
Error típico	0,0432	0,0294	0,0227	0,0223	0,0314	0,0228
Mediana	-0,1075	0,111	-0,1015	0,0445	-0,0950	0,0530
Moda	-	-	-	-	-	-
Desviación estándar	0,1498	0,1018	0,0786	0,0773	0,1089	0,0789
Varianza de la muestra	0,0224	0,0104	0,0062	0,006	0,0119	0,0062
Curtosis	-0,68	-1,27	0,18	-1,40	-1,12	0,17
Coefficiente de asimetría	-0,13	-0,57	0,75	-0,24	0,48	0,04
Rango	0,485	0,2960	0,264	0,231	0,328	0,289

Mínimo	0,134	-0,109	-0,185	0,141	-0,198	-0,091
Suma	-0,964	0,776	-1,012	0,518	-0,752	0,568
Cuenta	12	12	12	12	12	12

Realizado por : GUAÑO Wilmer, 2017

En la gráfica 1-3 se observa el porcentaje del error de las lecturas después del haber calibrado cada sensor que se ocupó para la implementación del prototipo.

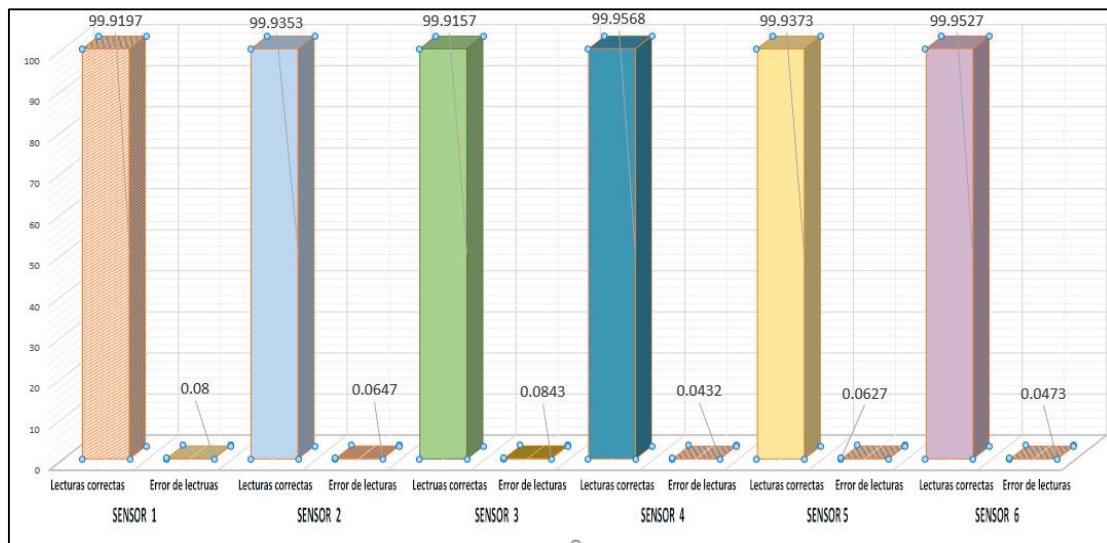


Gráfico 1-3: Pruebas de error de los sensores después de la calibración

Realizado por : GUAÑO Wilmer, 2017

Para la comprobación de nuestros coeficientes de correlación de cada uno de los sensores; realizado en referencia al anexo J, indicando la escala para que exista una buena correlación.

Los valores obtenidos anteriormente indican la obtención de una correlación positiva muy alta para cada sensor es decir que existe poca variabilidad en las lecturas de cada sensor de flujo y alcanzando la confiabilidad requerida.

3.7 Prueba de la Comunicación del prototipo

Para la comprobación de la comunicación se realizó en dos partes: la primera durante la implementación de cada módulo y la segunda cuando el prototipo se encontraba implementado en su totalidad. Para tener un correcto y adecuado funcionamiento de los dispositivos se realizó modificaciones al diseño con el fin de cumplir los requisitos planteados.

Para lo cual se utilizó comunicación RF para el envío de datos entre los dos módulos con la finalidad de abaratar el costo de implementación del prototipo ya que los datos son enviados de forma continua, se implementó comunicación GSM en el módulo de Recolección, Visualización y Control la cual permite envío de mensaje de alerta al personal encargado solo cuando exista una falla o fuga de agua potable en la red local de agua potable monitorizada

A demás el módulo de Recolección, Visualización y Control está conectado con un router mediante comunicación ethernet permitiendo el envío de información hacia la base de datos y accesibilidad a la página web.

3.7.1 Prueba de transmisión de datos

Se utilizó la pantalla del computador para comprobar la comunicación entre los módulos. Después de realizar las pruebas a cada nodo que conforma el módulo Lectura, Transmisión y Activación se confirmó el correcto funcionamiento entre los dos módulos. La comunicación se realizó a una distancia de 8m, la figura 6-3 se muestra la comunicación mediante RF.



Figura 6-3: Prueba de transmisión de datos
Realizado por : GUAÑO Wilmer, 2017

La figura 7-3 indica los datos enviados por los nodos que se encuentran en la red local de agua potable, estos se pueden observar en la pantalla Serial de Arduino, permitiendo confirmar la comunicación entre los dos módulos.

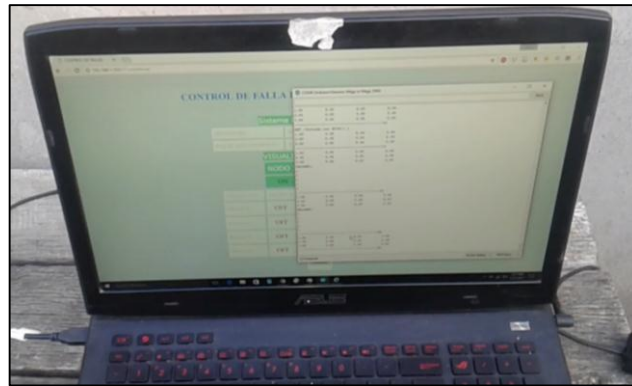


Figura 7-3: Prueba de transmisión en el observado en el computador
Realizado por : GUAÑO Wilmer, 2017

Para poder tener un óptimo desempeño de nuestro prototipo se configuró con un carácter que permite ordenar los datos de los módulos Lectura, Transmisión y Activación en forma descendente de cada uno evitando una confusión de los datos. Esto permite conocer que nodo no se encuentra conectado. En la tabla 8-3 como se envía la trama de datos.

Tabla 8-3: Cadena de caracteres de información de censado

Numero de nodo	Datos del sensor A	Datos del sensor B	Estado de la electroválvula A	Estado de la electroválvula B
1, 2,3...	Caudal (L/min)	Caudal (L/min)	Encendido / Apagado	Encendido / Apagado

Realizado por : GUAÑO Wilmer, 2017

3.7.2 Prueba de transmisión control a los nodos

Para poder tener un óptimo desempeño de nuestro prototipo se configuro con un carácter que permite recibir los datos del módulo Recolección, Visualización y Control. Esto permite

conocer el estado de activación de las electroválvulas. En la tabla 9-3 como se envía la trama de datos.

Tabla 9-3: Cadena de caracteres de información de control

Control de la electroválvula 1	Control de la electroválvula 2	Control de la electroválvula 3	Control de la electroválvula 4
Encendido / Apagado	Encendido / Apagado	Encendido / Apagado	Encendido / Apagado

Realizado por : GUAÑO Wilmer, 2017

3.7.3 Prueba de transmisión de alerta

Se implementó un módulo de comunicación de alerta el cual se muestra en la figura 8-3. Este se encarga de enviar un mensaje de texto mediante GSM al personal encargado del monitoreo de este sistema.



Figura 8-3: Prueba de comunicación mediante GSM

Realizado por : GUAÑO Wilmer, 2017

En el celular se puede observar en mensaje recibido, indicando que existen problemas en la red de agua potable. Lo cual permite conocer en tiempo real cuando algún ramal de la red de agua tenga alguna falla de funcionamiento.

3.8 Prueba de comunicación con la base de datos local

Para las pruebas de comunicación con la base de datos se divido en dos etapas: la primera es él envío de datos del Arduino el cual está programado según el Anexo k a un archivo (Anexo L) ubicado en las carpetas de la base de datos como se muestra en la figura 9-3, el archivo está programado con una extensión .PHP, la cual permite enlazar a los datos del Arduino con la base de datos.

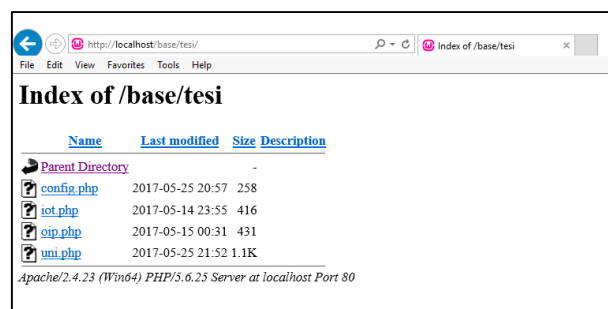


Figura 9-3: Archivos de configuración de enlace a la base de datos
Realizado por : GUAÑO Wilmer, 2017

La segunda parte consta la configuración de los archivos .PHP para esto es necesario información propia de la base de datos, la tabla 10-3 permite visualizar la configuración.

Tabla 10-3: Configuración de para enlace con la base de datos

Variable en lenguaje php	asignación
\$dbhost	"localhost";
\$dbuser	"proyecto2";
\$dbpass	"123456";
\$dbname	"control";

Realizado por : GUAÑO Wilmer, 2017

Las pruebas se realizaron utilizando la visualización del serial de Arduino y de la base de datos Wampserver64 como se observa en la figura 9-3.

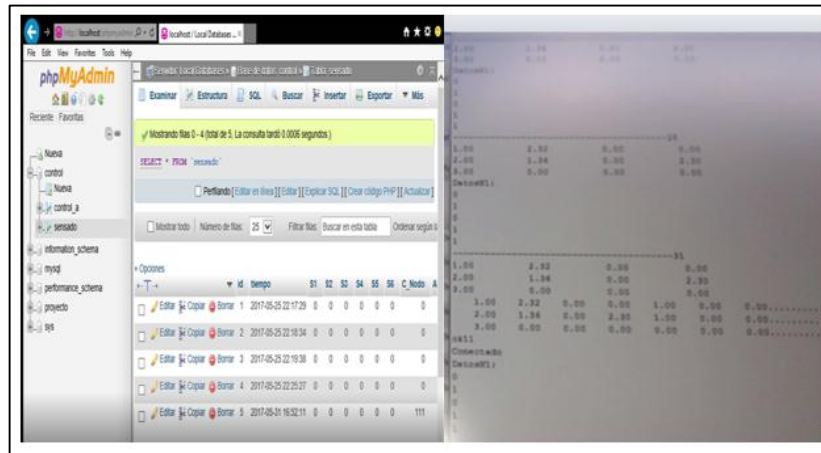


Figura 10-3: Fotografía de la comunicación entre la base de datos y Arduino
Realizado por : GUAÑO Wilmer, 2017

Se puede observar en la figura 10-3 que posee las variables necesarias de almacenamiento para cada sensor, además se ubicó una variable time la cual autogenera la hora y fecha que recibe los datos.

3.9 Prueba de comunicación con la página web

Las pruebas de comunicación realizadas entre el módulo y la herramienta desarrollada en formato HTML fueron similares a las realizadas en los distintos módulos. Debido a los procesos idénticos de comprobación mediante la visualización de la página web y el serial de Arduino. La figura 11-3 muestra las pruebas realizadas.



Figura 11-3: Pruebas de comunicación con la página web
Realizado por : GUAÑO Wilmer, 2017

3.10 Prueba de alimentación

Para realizar las pruebas de alimentación del prototipo se ha realizado una tabla en la cual se detalla el consumo de energía que posee cada dispositivo, se puede observar todos los datos recolectados en la tabla 11-3.

Tabla 11-3: Potencia de Consumo del prototipo

Potencia de consumo del prototipo				
Módulos del prototipo	Dispositivo	Voltaje (V)	Corriente (A)	Potencia (W)
Lectura- Transmisión- Activación	Tarjeta de desarrollo Arduino Nano	5	0.040	0.2
	Sensor YF-S201	5	0.015	0.075
	Electroválvula	12	0.320	3.84
	Módulo NRF24L01	3.3	0.014	0.0462
	Potencia Consumo del Módulo Lectura, Transmisión y Activación			
Recolección- Visualización- Control	Tarjeta de desarrollo Arduino Mega	5	0.040	0.2
	SIM800L	5	0.070	0.35
	Módulo de Ethernet shield	5	0.150	0.75
	Módulo NRF24L01	3.3	0.014	0.0462
	Potencia Consumo del Módulo Recolección, Visualización y Control			
POTENCIA TOTAL DE CONSUMO DEL PROTOTIPO				5.5074

Realizado por : GUANO Wilmer, 2017

La potencia que utiliza el módulo Lectura, Transmisión y Activación es de 4.1612 W con una alimentación de 5V, 0.8243 A, dividida en dos circuitos de alimentación uno para la parte de control con 0.3212 W voltaje de suministrado de 5 v, corriente de suministro de 0.0624 A y otra para la parte de potencia. El consumo de potencia que utiliza el Módulo Recolección,

Visualización y Control es de 1.3462 W con un voltaje de suministrado de 5 v y una corriente de suministro de 0.26924 A.

Para la alimentación del circuito de control del módulo de Lectura, Transmisión y Activación se utilizó una batería litio permitiendo una alimentación continua aproximada de 13 horas por parte de la batería cuando no exista presencia de luz solar, cálculo realizado por ecuación 3-1.

$$\text{vida utilil de la bateria} = \frac{(\text{capacidad de la bateria})}{\text{consumo del dispositivo}} * 0.70 \quad \text{Ecuación 3-1}$$

El factor de 0.7 permite tolerancias a factores externos que puedan afectar la vida útil de la batería.

Cuando exista presencia de luz solar el panel se encuentra conectado a un módulo tp 4056 el cual posee una eficiencia de 80% hacia la batería litio. La cual permite carga en 15 h. Cálculo realizado en la ecuación 3-2.

$$\text{numero de horas de carga} = \frac{(\text{capacidad de la bateria})}{\text{amperaje de carga del panel solar}*(0.80)} \quad \text{Ecuación 3-2}$$

3.11 Implementación del prototipo

Después de todas las pruebas realizadas a los diferentes dispositivos y la calibración de los sensores que permiten la recolección de datos para su posterior control, se procede a la puesta en funcionamiento del prototipo, supervisando los datos obtenidos mediante la herramienta virtual de visualización que fue desarrollada en HTML, se verifico el correcto funcionamiento de la comunicación entre los dos módulos, comprobándose que los datos de monitoreo y control sean transmitidos correctamente, a su vez la comprobación de activación de las electroválvulas. En la figura 12-3 se puede observar la fotografía antes de conectar a la red de agua potable, y la figura 13-3 muestra una fotografía de todo el prototipo implementado.



Figura 12-3: Módulos antes de conectar a la red de agua potable
Realizado por : GUAÑO Wilmer, 2017

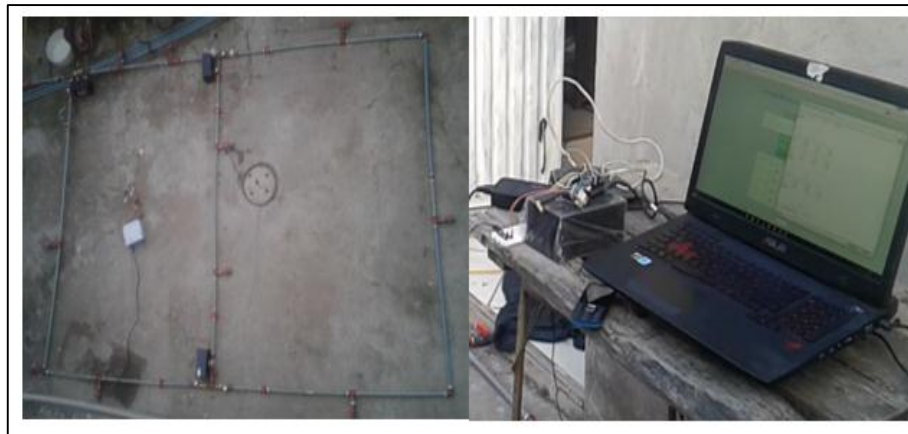


Figura 13-3: Implementación del prototipo en la red local de agua potable
Realizado por : GUAÑO Wilmer, 2017

3.12 Análisis económico del prototipo

Para la implementación del prototipo de monitoreo de flujo de agua mediante GSM y radio frecuencia en un circuito local de agua potable se realizó un análisis económico de la implementación, con elementos adquiridos en el Ecuador. En este análisis no se incluyen los costos estructurales que necesitan cada módulo y la red de agua potable por considerarla fuera de alcance de evaluación del prototipo

Tabla 12-3: Análisis económica del prototipo

Modulas de prototipo	Componentes	Cantidad	Costo (USD)
Lectura-Transmisión- Activación	Tarjeta Arduino NANO	3	21
	Sensor YF-S201	6	36
	Electroválvula	4	60
	NRF24L01	3	22.5
	Panel solar 9v	3	18
	Módulo tp4056	3	9
	Batería de litio de 3.7v y 1200mA	3	45
	Cables para la interconexión	1	7
	TIP 121	4	4
	Relé 5v	4	3
	Costo del Módulo Lectura, Transmisión y Activación		
Recolección- Visualización-Control	Tarjeta Arduino MEGA	1	35
	Shield Ethernet	1	25
	SIM800L	1	30
	NRF24L01	1	7,50
	Cables para la interconexión	1	7
	Costo del Módulo Recolección, Visualización y Control		
Costo total del prototipo			270

Realizado por : GUAÑO Wilmer, 2017

En la tabla 12-3 se presenta el listado de los elementos y costos para cada uno de los módulos que integran el prototipo. El módulo Lectura, Transmisión y Activación presenta un costo de 165.5 USD superior al módulo Recolección, Visualización y Control con un costo de 104.5 USD por motivo que se analiza componentes de nodos es decir cada nodo tendría un costo 55.17 USD.

El costo es bajo en comparación a la inestabilidad del servicio de agua potable, ya que cuando existe una falla de agua potable no posee una inmediata solución a la problemática generando perdida de líquido vital, sin contar que en el mercado no existe un sistema similar al implementado que brinda información precisa ayudando a la empresa y al usuario con una rápida solución a los problemas que se presentan ante una falla de distribución de agua potable.

CONCLUSIONES

- Se implementó un prototipo de bajo costo analizando los diferentes elementos, en función a las características de cada dispositivo utilizado en los dos módulos que lo conforman.
- El prototipo dispone de 4 electroválvulas las cuales permiten el control del abastecimiento para cada ramal en el circuito local de agua potable, abriendo cuando el sistema no detecte fuga o falla en la red y cerrando cuando exista fallas en cada ramal de la red de agua potable.
- En la programación desarrollada en la plataforma Arduino se realizó mediante funciones o módulos permite establecer con claridad los procesos de funcionamiento del prototipo.
- Se incorporó un sistema de comunicación de RF entre el módulo Lectura, Transmisión y Activación y el módulo Recolección, Visualización y Control que permite la transmisión y recepción de datos censados para control, comprobación y activación de las electroválvulas. Se incorporó también un módulo que permite la comunicación GSM el cual envía un mensaje de texto al personal encargado, informándole del estado de la red local de agua potable.
- La implementación de la página web se realizó en HTML, donde se muestran los valores medidos por cada sensor utilizado. Dispone de opciones mediante botones que permiten activar uno o varios ramales para el proceso de control de la red local de agua potable. Además, permite visualizar el estado de cada electroválvula.
- De las pruebas realizadas en cada uno de los sensores se logró determinar que el prototipo presenta un error de monitoreo diferente en cada sensor de 0.051733 % que corresponde un flujo de 0.571 L/min en un total de 10 L/min.

RECOMENDACIONES

- Para usos de tiempo prolongados en el prototipo se recomienda incorporar una alimentación autónoma para el circuito de potencia.
- Mejoramiento de la estructura física del prototipo para evitar filtraciones de agua a los diferentes módulos.
- Para una comunicación a mayores a 40 m en RF, desde un nodo hasta el punto base se debe utilizar un módulo de comunicación con mejores prestaciones y alcance.
- La ubicación de los sensores se debe realizar siempre en el mismo sentido que el flujo de agua.
- Hacer un control periódico preventivo a los dispositivos del prototipo para un mejor funcionamiento del mismo.
- Hacer una calibración de los sensores de flujo de agua antes de su implementación.

BIBLIOGRAFÍA

1. **AMANGANDI, Jonathan.** *Arduino* [blog]. 2012. [Citado el: 17 de abril de 2017]. Disponible en: <http://jamangandi2012.blogspot.com/2012/10/que-es-arduino-te-lo-mostramos-en-un.html>.
2. **BLÁZQUEZ, J.** *Introducción a los Sistemas de Comunicación Inalámbrica*. Catalunya-España: Eureka Media, 2013. pp. 7-21.
3. **CALETTI, Eduardo.** *Conceptos generales Descripción y funcionamiento*. Buenos Aires. 2017. [Citado el: 21 de abril de 2017]. Disponible en: <http://robots-argentina.com.ar/robots-nos.htm>
4. **CANTILLO MAROTO, Sebastián.** *Desarrollo de aplicaciones basadas en WSN*. (TESIS). (Pregrado) Universidad Politécnica de Valencia. Escuela Superior de Ingeniería Informática. Valencia España. 2010 pp. 8, 22, 16, 12. Disponible en: <https://riunet.upv.es/bitstream/handle/10251/8592/PFC%20-%20DESARROLLO%20DE%20APLICACIONES%20BASADAS%20EN%20WSN.pdf>. [Citado el: 21 de abril de 2017].
5. **CCM BENCHMARK.** “Introducción a base de datos”. *CCM BENCHMARK*. [En línea]. 2017. pp. 1-2 [Citado el: 13 de mayo de 2017]. Disponible en: <http://es.ccm.net/contents/81-el-equipode-ccm-en-espanol>
6. **CESAR.** *Ayuda Electrónica | Circuitos Electrónicos, Ingeniería Electronica*. 2010. [Citado el: 20 de abril de 2017]. Disponible en: <http://ayudaelectronica.com/%C2%BFque-son-los-arduino-shields/>.
7. **CIANCA, Kiara.** *Las redes con sensores*. [blog]. [Citado el: 10 de abril de 2017]. Disponible en: http://lasredesconsensores.blogspot.com/2012_07_01_archive.html
8. **CULTURACION.** *Raspberry: Pi que es características precios*. [blog]. 2016. [Citado el: 17 de abril de 2017]. Disponible en: <http://culturacion.com/raspberry-pi-que-es-caracteristicas-y-precios/>
9. **DOME, E.** “Radiofrecuencias y Microondas”. *Centro Científico Tecnológico (CCT) CONICET Mendoza*. [En línea], 2017. [Citado el: 18 de abril de 2017]. Disponible en: <http://www.cricyt.edu.ar/enciclopedia/terminos/RadioyMicro.htm>
10. **GASCÓN, D.** “Redes de Sensores inalámbricos, la tecnología invisible”. Libelium. Disponible en: <http://www.libelium.com/libelium-downloads/libelium-bit-coit.pdf>. 2010. pp. 53. [Citado el: 12 de abril de 2017.]

11. **GÓMEZ, Guillermo.** *Fugas de agua en redes de agua potable.* FugaTec [blog].2015. [Citado el: 10 de mayo de 2017]. Disponible en: <https://fugatec.com/blog/fugas-de-agua-en-redes-de-agua-potable/>
12. **GUSQUI MACAS, Gloria. R & CHAGLLA MOINA, Ana. G.** *Implementación de un sistema electrónico inteligente de asignación, medición y alerta del consumo de agua potable para hogares residenciales de la ciudad de Riobamba* (TESIS). (Pregrado) Escuela Superior Politécnica de Chimborazo, Facultad de Informática y Electrónica, Riobamba Ecuador, 2016.
13. **LLANGA HERRERA, Darío; & CANO ESTRELLA, David.** *Implementación del prototipo para el control de acceso docente mediante la interoperabilidad de WSN y RFID.* (TESIS). (Pregrado) Escuela Superior Politécnica de Chimborazo, Facultad de Informática y Electrónica, Riobamba Ecuador, 2016 pp. 39.
14. **MATURANA, Jesús.** *Intel Galileo, placa de desarrollo fruto de la colaboración entre Intel y Arduino.* [blog].2013. [Citado el: 17 de abril de 2017]. Disponible en: <https://www.xataka.com/makers/intel-galileo-placa-de-desarrollo-fruto-de-la-colaboracion-entre-intel-y-arduino>.
15. **ORGANIZACION DE LAS NACIONES UNIDAS PARA ALIMENTOS Y AGRICULTURA(FAO).** *Naciones ambientales básicas para profesores rurales y extensionistas.* [En línea]. Roma. Italia 2015. [Citado el: 07 de abril de 2017]. Disponible en: <http://www.fao.org/docrep/006/w1309s/w1309s06.htm#TopOfPage>.
16. **ORGANIZACION DE LAS NACIONES UNIDAS PARA LA EDUCACION, CIENCIA Y CULTURA(UNESCO).**” *Agua para todos Agua para la vida*”. Paris. Francia Mundi-Prensa Libros.2005. [En línea]. [Citado el: 10 de abril de 2017]. Disponible en: <http://unesdoc.unesco.org/images/0012/001295/129556s.pdf>. pp.4.
17. **QUADRATO.** *Características básicas de un sitio web bien desarrollado.* [blog]. México. 2017. [Citado el: 15 de mayo de 2017]. Disponible en: <http://blog.quadrato.mx/caracteristicas-basicas-de-un-sitio-web-bien-desarrollado/>
18. **QUEES.** *página web.* 2017. [Citado el: 21 de abril de 2017]. Disponible en: <http://www.quees.info/que-es-una-pagina-web.html>
19. **ROMO AZUNA, Raúl.** *Procesos constitutivos para obras de aprovisionamiento de agua potable.* [En línea]. México.1991. Capítulo 6 pp 1-2. [Citado el: 09 de abril de 2017]. Disponible en: <http://www.bidi.uson.mx/tesisDet.aspx?crit1=IDUT&texto1=5618&crit2=TITULO&texto2=>
20. **SALGADO CAZORLA, Cristian. A & CARRANZA CHICHANDE, Geomayra.** *Diseño de un prototipo para la implementación de un sistema de monitoreo de vibración e*

- inclinación de un puente mediante una red de sensores inalámbricos.* (TESIS). (Pregrado) Escuela Superior Politécnica de Chimborazo, Facultad de Informática y Electrónica, Riobamba Ecuador, 2016 pp. 5.
21. **SIDOROVA, A.** *Aplicación del método de la emisión acústica en la monitorización de las estructuras de hormigón.* (TESIS). (Pregrado) Universidad Politécnica de Cataluña, Ingeniería Estructural y de la Construcción, Barcelona - España, 2010, pp. 18-21.
 22. **TAPIA ORTIZ, Francisco.** “Redes de sensores inalámbricos”. *Universidad Técnica Federico Santa María.* [En línea]. 2008 Chile. pp. 8. [Citado el: 23 de abril de 2017]. Disponible en: http://profesores.elo.utfsm.cl/~tarredondo/info/networks/Presentacion_sensores.pdf
 23. **TECBOLIVIA.** *Intel Galileo.* Bolivia: 2017. [Citado el: 15 de mayo de 2017]. Disponible en: <http://tecbolivia.com/index.php/venta-de-componentes-electronicos-11/arduino/intel-galileo-detail>
 24. **UNIVERSIDAD POLITÉCNICA DE VALENCIA.** *Historia de la informática.* [blog].2013. [Citado el: 12 de mayo de 2017]. Disponible en: <http://histinf.blogs.upv.es/2013/12/18/raspberry-pi/>
 25. **VALDÉS PÉREZ, Damián.** *Los diferentes lenguajes de programación para la web.* Maestros de la Web. 2007. [Citado el: 16 de mayo de 2017]. Disponible en: <http://www.maestrosdelweb.com/los-diferentes-lenguajes-de-programacion-para-la-web/>
 26. **VARONAS, Nico.** *Galileo: La placa de desarrollo de Intel y Arduino.* 2013. [Citado el: 17 de abril de 2017]. Disponible en: <http://www.neoteo.com/galileo-la-placa-de-desarrollod-de-intel-y-arduino/>.

ANEXOS

Anexo A: Módulo de comunicación NRF24L01

PRELIMINARY PRODUCT SPECIFICATION **NORDIC SEMICONDUCTOR**

Single chip 2.4 GHz Transceiver **nRF24L01**

FEATURES

- True single chip GFSK transceiver
- Complete OSI Link Layer in hardware
- Enhanced ShockBurst™
- Auto ACK & retransmit
- Address and CRC computation
- On the air data rate 1 or 2Mbps
- Digital interface (SPI) speed 0-8 Mbps
- 125 RF channel operation
- Short switching time enable frequency hopping
- Fully RF compatible with nRF24XX
- 5V tolerant signal input pads
- 20-pin package (QFN20 4x4mm)
- Uses ultra low cost +/- 60 ppm crystal
- Uses low cost chip inductors and 2-layer PCB
- Power supply range: 1.9 to 3.6 V

APPLICATIONS

- Wireless mouse, keyboard, joystick
- Keyless entry
- Wireless data communication
- Alarm and security systems
- Home automation
- Surveillance
- Automotive
- Telemetry
- Intelligent sports equipment
- Industrial sensors
- Toys

GENERAL DESCRIPTION
 nRF24L01 is a single chip radio transceiver for the world wide 2.4 - 2.5 GHz ISM band. The transceiver consists of a fully integrated frequency synthesizer, a power amplifier, a crystal oscillator, a demodulator, modulator and Enhanced ShockBurst™ protocol engine. Output power, frequency channels, and protocol setup are easily programmable through a SPI interface. Current consumption is very low, only 9.0mA at an output power of -6dBm and 12.3mA in RX mode. Built-in Power Down and Standby modes makes power saving easily realizable.

QUICK REFERENCE DATA

Parameter	Value	Unit
Minimum supply voltage	1.9	V
Maximum output power	0	dBm
Maximum data rate	2000	kbps
Supply current in TX mode @ 0dBm output power	11.3	mA
Supply current in RX mode @ 2000 kbps	12.3	mA
Temperature range	-40 to +85	°C
Sensitivity @ 1000 kbps	-85	dBm
Supply current in Power Down mode	900	nA

Table 1 nRF24L01 quick reference data

Nordic Semiconductor ASA · Vestre Rosten 81, N-2075 Tiller, Norway · Phone +4772898900 · Fax +4772898999
 Revision: 1.2 Page 1 of 39 March 2006

PRELIMINARY PRODUCT SPECIFICATION **NORDIC SEMICONDUCTOR**

nRF24L01 Single Chip 2.4 GHz Radio Transceiver

Type Number	Description	Version
nRF24L01	20 pin QFN 4x4, RoHS & SS-00259 compliant	D
nRF24L01 IC	Bare Dice	D
nRF24L01-EVKIT	Evaluation kit (2 test PCB, 2 configuration PCB, SW)	1.0

Table 2 nRF24L01 ordering information

BLOCK DIAGRAM

Figure 1 nRF24L01 with external components.

Nordic Semiconductor ASA · Vestre Rosten 81, N-2075 Tiller, Norway · Phone +4772898900 · Fax +4772898999
 Revision: 1.2 Page 36 of 39 March 2006

PRELIMINARY PRODUCT SPECIFICATION **NORDIC SEMICONDUCTOR**

nRF24L01 Single Chip 2.4 GHz Radio Transceiver

PIN FUNCTIONS

Pin	Name	Pin function	Description
1	CE	Digital Input	Chip Enable Activates RX or TX mode
2	CSN	Digital Input	SPI Chip Select
3	SCK	Digital Input	SPI Clock
4	MOSI	Digital Input	SPI Slave Data Input
5	MISO	Digital Output	SPI Slave Data Output, with tri-state option
6	IRQ	Digital Output	Maskable interrupt pin
7	VDD	Power	Power Supply (+3V DC)
8	VSS	Power	Ground (0V)
9	XCI	Analog Output	Crystal Pin 2
10	XCP	Analog Output	Crystal Pin 1
11	VDD_PA	Power Output	Power Supply (+1.8V) to Power Amplifier
12	ANT1	RF	Antenna interface 1
13	ANT2	RF	Antenna interface 2
14	VSS	Power	Ground (0V)
15	VDD	Power	Power Supply (+3V DC)
16	DREF	Analog Input	Reference current
17	VSS	Power	Ground (0V)
18	VDD	Power	Power Supply (+3V DC)
19	DVDD	Power Output	Positive Digital Supply output for decoupling purposes
20	VSS	Power	Ground (0V)

Table 3 nRF24L01 pin function

PIN ASSIGNMENT

Figure 2 nRF24L01 pin assignment (top view) for a QFN20 4x4 package.

Nordic Semiconductor ASA · Vestre Rosten 81, N-2075 Tiller, Norway · Phone +4772898900 · Fax +4772898999
 Revision: 1.2 Page 3 of 39 March 2006

PRELIMINARY PRODUCT SPECIFICATION **NORDIC SEMICONDUCTOR**

nRF24L01 Single Chip 2.4 GHz Radio Transceiver

Figure 15 Module with OFM crystal and SMA connector

Nordic Semiconductor ASA · Vestre Rosten 81, N-2075 Tiller, Norway · Phone +4772898900 · Fax +4772898999
 Revision: 1.2 Page 36 of 39 March 2006

PRELIMINARY PRODUCT SPECIFICATION



nRF24L01 Single Chip 2.4 GHz Radio Transceiver

ELECTRICAL SPECIFICATIONS

Conditions: VDD = +3V, VSS = 0V, TA = -40°C to +85°C

Symbol	Parameter (condition)	Notes	Min.	Typ.	Max.	Units
Operating conditions						
VDD	Supply voltage		1.9	3.0	3.6	V
TEMP	Operating Temperature		-40	+27	+85	°C
Digital input pin						
V _{IH}	HIGH level input voltage	¹	0.7VDD		5.25	V
V _{IL}	LOW level input voltage		VSS		0.3VDD	V
Digital output pin						
V _{OH}	HIGH level output voltage (I _{OL} =-0.25mA)		VDD-0.3		VDD	V
V _{OL}	LOW level output voltage (I _{OH} =0.25mA)		VSS		0.3	V
General RF conditions						
f _{op}	Operating frequency	²	2400		2525	MHz
f _{crystal}	Crystal frequency			16		MHz
Δf _{100k}	Frequency deviation @ 1000kpbs			±160		kHz
Δf _{200k}	Frequency deviation @ 2000kpbs			±320		kHz
R _{data}	Data rate ShockBurst™		50		2000	kbps
F _{channel}	Channel spacing @ 1000kpbs			1		MHz
F _{channel}	Channel spacing @ 2000kpbs			2		MHz
Transmitter operation						
P _{max}	Maximum Output Power		0	+4		dBm
P _{1dB}	RF Power Control Range		16	18	20	dB
P _{acc}	RF Power Accuracy				±4	dB
BW	20dB Bandwidth for Modulated Carrier (2000kpbs)		1800		2000	kHz
P _{1dB}	1 st Adjacent Channel Transmit Power 2MHz				-20	dBm
P _{2dB}	2 nd Adjacent Channel Transmit Power 4MHz				-50	dBm
I _{cc}	Supply current @ 0dBm output power			11.3		mA
I _{cc}	Supply current @ -18dBm output power			7.0		mA
I _{cc}	Average supply current @ -50dBm output power, Enhanced ShockBurst™			0.05		mA
I _{cc}	Supply current in Standby mode			32		µA
I _{cc}	Supply current in power down			900		nA

¹ All digital inputs handle up to 5.25V signal inputs. Keep in mind that the VDD of the nRF24L01 must match the V_{DD} of the driving device for output pins.
² Usable band is determined by local regulations.
³ Antenna load impedance = 15Ω±j88Ω
⁴ Antenna load impedance = 15Ω±j88Ω. Effective data rate 1000kpbs or 2000 kpbs
⁵ Antenna load impedance = 15Ω±j88Ω. Effective data rate 10kpbs and full packets
⁶ Given for a 12pF crystal. Current when using external clock is dependent on signal swing.

PRELIMINARY PRODUCT SPECIFICATION



nRF24L01 Single Chip 2.4 GHz Radio Transceiver

Receiver operation						
I _{cc}	Supply current one channel 2000kpbs			12.3		mA
I _{cc}	Supply current one channel 1000kpbs			11.8		mA
R _{Xsens}	Sensitivity at 0.1%BER (@ 2000kpbs)			-82		dBm
R _{Xsens}	Sensitivity at 0.1%BER (@ 1000kpbs)			-83		dBm
C _{I_{cc}}	C/I Co-channel (@ 2000kpbs)			77/1 ¹		dB
C _{I_{cc}}	1 st Adjacent Channel Selectivity C/I 2MHz			14		dB
C _{I_{cc}}	2 nd Adjacent Channel Selectivity C/I 4MHz			-216/20		dB
C _{I_{cc}}	1 st Adjacent Channel Selectivity C/I 6MHz			-274/27		dB
C _{I_{cc}}	C/I Co-channel (@ 1000kpbs)			97/12 ²		dB
C _{I_{cc}}	1 st Adjacent Channel Selectivity C/I 1MHz			86		dB
C _{I_{cc}}	2 nd Adjacent Channel Selectivity C/I 2MHz			-22/21		dB
C _{I_{cc}}	1 st Adjacent Channel Selectivity C/I 3MHz			-306/30		dB

Table 4 nRF24L01 RF specifications

¹ Data rate is 2000kpbs for the following C/I measurements
² According to ETSI EN 300 440-1 V1.3.1 (2001-09) page 27
³ nRF24L01 equal modulation on interfering signal
⁴ Data rate is 1000kpbs for the following C/I measurements
⁵ According to ETSI EN 300 440-1 V1.3.1 (2001-09) page 27
⁶ nRF24L01 equal modulation on interfering signal

PRELIMINARY PRODUCT SPECIFICATION



nRF24L01 Single Chip 2.4 GHz Radio Transceiver

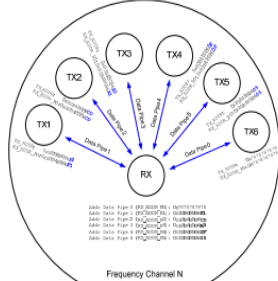


Figure 5: Example on how the acknowledgement addressing is done

An nRF24L01 configured as PTX with Enhanced ShockBurst™ enabled, will use the ShockBurst™ feature to send a packet whenever the microcontroller wants to. After the packet has been transmitted, nRF24L01 will switch on its receiver and expect an acknowledgement to arrive from the terminating part. If this acknowledgement fails to arrive, nRF24L01 will retransmit the same packet until it receives an acknowledgement or the number of retries exceeds the number of allowed retries given in the SETUP_RETR_ARC register. If the number of retries exceeds the number of allowed retries, this will be showed by the STATUS register bit MAX_RT which gives an interrupt.

Whenever an acknowledgement is received by an nRF24L01 it will consider the last transmitted packet as delivered. It will then be cleared from the TX FIFO, and the TX_DS IRQ source will be set high.

With Enhanced ShockBurst™ nRF24L01 offers the following benefits:

- Highly reduced current consumption due to short time on air and sharp timing when operating with acknowledgement traffic
- Lower system cost. Since the nRF24L01 handles all the high-speed link layer operations, like re-transmission of lost packet and generating acknowledgement to received packets, it is no need for hardware SPI on the system microcontroller to interface the nRF24L01. The interface can be done by using general purpose IO pins on a low cost microcontroller where the SPI is emulated in firmware. With the nRF24L01 this will be sufficient speed even when running a bi-directional link.
- Greatly reduced risk of "on-air" collisions due to short time on air
- Easier firmware development since the link layer is integrated on chip

PRELIMINARY PRODUCT SPECIFICATION



nRF24L01 Single Chip 2.4 GHz Radio Transceiver

DEVICE CONFIGURATION

All configuration of nRF24L01 is defined by values in some configuration registers. All these registers are writable via the SPI interface.

SPI Interface

The SPI interface is a standard SPI interface with a maximum data rate of 10Mbps. Most registers are readable.

SPI Instruction Set

The available commands to be used on the SPI interface are shown below. Whenever CSN is set low the interface expects an instruction. Every new instruction must be started by a high to low transition on CSN.

In parallel to the SPI instruction word applied on the MOSI pin, the STATUS register is shifted serially out on the MISO pin.

The serial shifting SPI commands is on the format:

<Instruction word>: MSBit to LSBit (one byte)
 <Data bytes>: LSBit to MSByte, MSBit in each byte first

See Figure 8 and Figure 9.

Instruction Name	Instruction Format [binary]	# Data Bytes	Operation
R_REGISTER	000A AAAA	1 to 5 LSByte first	Read registers. AAAAA = 5 bit Memory Map Address
W_REGISTER	001A AAAA	1 to 5 LSByte first	Write registers. AAAAA = 5 bit Memory Map Address <i>Executable in power down or standby modes only.</i>
R_RX_PAYLOAD	0110 0001	1 to 32 LSByte first	Read RX-payload: 1 = 32 bytes. A read operation will always start at byte 0. Payload will be deleted from FIFO after it is read. Used in RX mode.
W_TX_PAYLOAD	1010 0000	1 to 32 LSByte first	Used in TX mode. Write TX-payload: 1 = 32 bytes. A write operation will always start at byte 0.
FLUSH_TX	1110 0001	0	Flush TX FIFO, used in TX mode
FLUSH_RX	1110 0010	0	Flush RX FIFO, used in RX mode Should not be executed during transmission of acknowledgement, i.e. acknowledge package will not be completed.
REUSE_TX_PL	1110 0011	0	Used for a PTX device Reuse last sent payload. Packets will be repeatedly resent as long as CE is high. TX payload reuse is active until W_TX_PAYLOAD or FLUSH_TX is executed. TX payload reuse must not be activated or deactivated during package transmission
NOP	1111 1111	0	No Operation. Might be used to read the STATUS register

Table 8 Instruction set for the nRF24L01 SPI interface.

The W_REGISTER and R_REGISTER may operate on single or multi-byte registers. When accessing multi-byte registers one will read or write MSBit of LSBit first. The



nRF24L01 Single Chip 2.4 GHz Radio Transceiver

APPLICATION EXAMPLE

nRF24L01 with single ended matching network crystal, bias resistor, and decoupling capacitors.

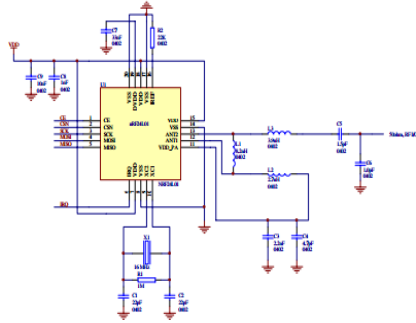


Figure 13 nRF24L01 schematic for RF layouts with single ended 50Ω RF output.

Part	Designator	Footprint	Description
22pF ¹⁷	C1	0402	NPO, +/- 2%, 50V
22pF ¹⁷	C2	0402	NPO, +/- 2%, 50V
2.2nF	C3	0402	X7R, +/- 10%, 50V
4.7nF	C4	0402	NPO, +/- 0.25 pF, 50V
1.5pF	C5	0402	NPO, +/- 0.1 pF, 50V
1.0pF	C6	0402	NPO, +/- 0.1 pF, 50V
33nF	C7	0402	X7R, +/- 10%, 50V
10nF	C8	0402	X7R, +/- 10%, 50V
10nF	C9	0402	X7R, +/- 10%, 50V
8.2nH	L1	0402	chip inductor +/- 5%
2.7nH	L2	0402	chip inductor +/- 5%
3.9nH	L3	0402	chip inductor +/- 5%
10k	R1	0402	+/-10%
22k	R2	0402	+/-1%
nRF24L01	U1	GFN20 4x4	
16MHz	X1		+/-60ppm, C ₀ =12pF ¹⁷

Table 16 Recommended components (BOM) in nRF24L01 with antenna matching network

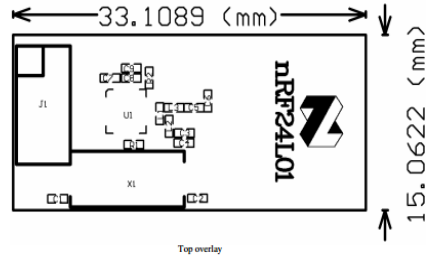
¹⁷ C1 and C2 must have values that match the crystals load capacitance, C_L.



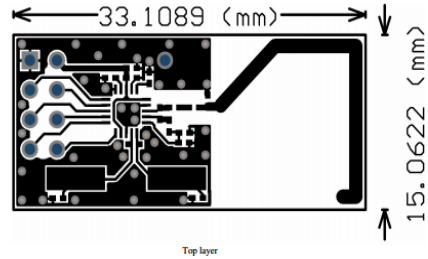
nRF24L01 Single Chip 2.4 GHz Radio Transceiver

PCB layout examples

Figure 14 shows a PCB layout example for the application schematic in Figure 13. A double-sided FR-4 board of 1.6mm thickness is used. This PCB has a ground plane on the bottom layer. Additionally, there are ground areas on the component side of the board to ensure sufficient grounding of critical components. A large number of via holes connect the top layer ground areas to the bottom layer ground plane.

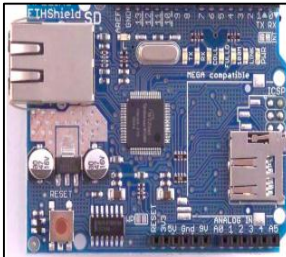


Top overlay



Top layer

Anexo B: Módulo shield Ethernet de Arduino



Download: [arduino-ethernet-shield-05-schematic.pdf](#), [arduino-ethernet-shield-05-reference-design.zip](#)

Download: [arduino-ethernet-shield-schematic.pdf](#), [arduino-ethernet-shield-reference-design.zip](#)

The Arduino Ethernet Shield allows an Arduino board to connect to the internet. It is based on the [Wiznet W5100](#) ethernet chip ([datasheet](#)). The Wiznet W5100 provides a network (IP) stack capable of both TCP and UDP. It supports up to four simultaneous socket connections. Use the [Ethernet library](#) to write sketches which connect to the internet using the shield. The ethernet shield connects to an Arduino board using long wire-wrap headers which extend through the shield. This keeps the pin layout intact and allows another shield to be stacked on top.

The latest revision of the shield adds a micro-SD card slot, which can be used to store files for serving over the network. It is compatible with the Arduino Duemilanove and Mega (using the Ethernet library coming in Arduino 0019). An SD card library is not yet included in the standard Arduino distribution, but the [sdfatlib](#) by Bill Greiman works well. See [this tutorial from Adafruit Industries](#) for instructions (thanks Limor!).

The latest revision of the shield also includes a reset controller, to ensure that the W5100 Ethernet module is properly reset on power-up. Previous revisions of the shield were not compatible with the Mega and need to be manually reset after power-up. The original revision of the shield contained a full-size SD card slot; this is not supported.

Arduino communicates with both the W5100 and SD card using the SPI bus (through the ICSP header). This is on digital pins 11, 12, and 13 on the Duemilanove and pins 50, 51, and 52 on the Mega. On both boards, pin 10 is used to select the W5100 and pin 4 for the SD card. These pins cannot be used for general I/O. On the Mega, the hardware SS pin, 53, is not used to select either the W5100 or the SD card, but it must be kept as an output or the SPI interface won't work.

Note that because the W5100 and SD card share the SPI bus, only one can be active at a time. If you are using both peripherals in your program, this should be taken care of by the corresponding libraries. If you're not using one of the peripherals in your program, however, you'll need to explicitly deselect it. To do this with the SD card, set pin 4 as an output and write a high to it. For the W5100, set digital pin 10 as a high output.

The shield provides a standard RJ45 ethernet jack.

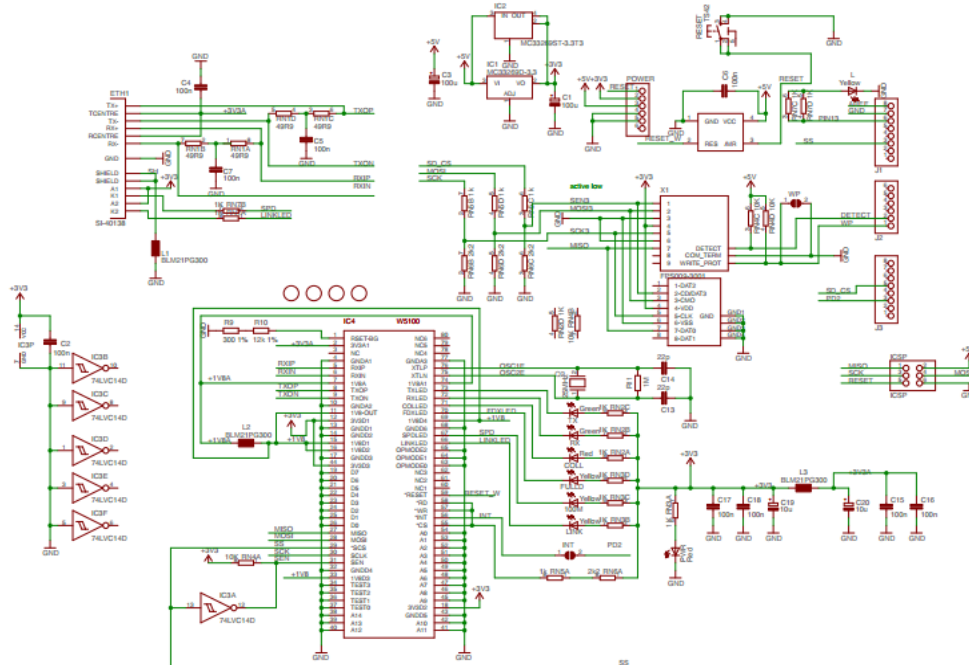
The reset button on the shield resets both the W5100 and the Arduino board.

The shield contains a number of informational LEDs:

- PWR: indicates that the board and shield are powered
- LINK: indicates the presence of a network link and flashes when the shield transmits or receives data
- FULLD: indicates that the network connection is full duplex
- 100M: indicates the presence of a 100 Mb/s network connection (as opposed to 10 Mb/s)
- RX: flashes when the shield receives data
- TX: flashes when the shield sends data
- COLL: flashes when network collisions are detected

The solder jumper marked "INT" can be connected to allow the Arduino board to receive interrupt-driven notification of events from the W5100, but this is not supported by the Ethernet library. The jumper connects the INT pin of the W5100 to digital pin 2 of the Arduino.

See also: [getting started with the ethernet shield](#) and [Ethernet library reference](#)



Arduino ETHERNET - shield V5

Copyright (c) 2010 Arduino
Released under the Creative Commons Attribution-Share Alike 3.0 License
<http://creativecommons.org/licenses/by-sa/3.0/>

Activate (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset on software running on a connected computer. One of the hardware pins (called reset) on the ATmega2560 is connected to the reset pin of the ATmega2560 via a 10K non-volatile capacitor. When this line is asserted (taken out), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a longer timeout, as the timing of DTR can be well-coordinated with the start of the upload.

The way the other instructions, when the Mega2560 is connected to either a computer running Mac OS X or Linux, it needs just one connection is made to it from software (via USB). For the following explanation of so, the bootloader is running on the Mega2560, while it is programmed to grab non-volatile data (i.e. anything besides an upload of new code). It will interrupt the first few cycles of data sent to the board after a connection is opened. It's a careful timing, as the board receives some information at other times when it first starts. Make sure that the software with which it communicates sets a second after opening the connection and before sending the data.

The Mega contains a fuse that can be set to disable the auto-reset. The pins on either side of the board can be soldered together to re-enable it. It's called "RSTTRST". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from D+ to the reset pin. See [this forum thread](#) for details.

USB Overcurrent Protection

The Arduino Mega has a hardware analyzer that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, it has provided an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection with the port or board is removed.

Physical Characteristics and Shield Compatibility

The maximum width and size of the Mega PCB are 4.0x12.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes along the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 190 mil (5.141"), not an even multiple of the 100 mil spacing of the other pins.

The Mega is designed to be compatible with most shields designed for the Duemilanove or Duemilanove. Digital pins 0 to 13 (and the adjacent AVCC and GND pins), analog pins 0 to 5, the power header, and USB header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1, pins 2 and 3 (respectively). SPI is available through the ICSP header on both the Mega and Duemilanove. (Warning: Please note that ICSP is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Duemilanove (pinning includes 4 and 5).

How to use Arduino



Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The microcontroller on the board is programmed using the [Arduino programming language](#) (based on C/C++) and the Arduino development environment (based on [Processing](#)). Arduino projects can be shared online as they can communicate with software on running on a computer (e.g. Flash, Processing, MATLAB).

Arduino is a cross-platform program, you'll have to follow different instructions for your personal OS. Check on the [Arduino site](#) for the latest instructions. [http://arduino.cc/en/GuideMainPage](#)

Linux Install

Windows Install

Mac Install

Once you have downloaded and installed the Arduino IDE, you can plug the Arduino to your PC via USB cable.

Quick Start

Now you're actually ready to "burn" your first program on the Arduino board. To avoid "burn out", the physical structure of the well known programming "burn out", MEGA

File > Sketchbook > Arduino (001) > Examples > Digital > Blink

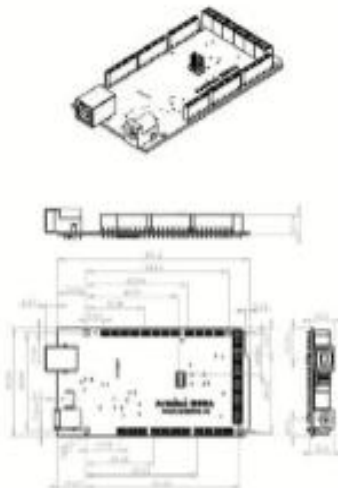
Once you have your sketch you'll see something very close to the screenshot on the right.

In Tools > Board select MEGA

Now you have to go to Tools > Serial Port and select the right serial port. The serial connector is attached to.



Dimensioned Drawings



Terms & Conditions



1. Introduction

1.1. The Arduino company (the "Company") is the proprietor of the Arduino IDE. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.2. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.3. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.4. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.5. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.6. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.7. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.8. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.9. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.10. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.11. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.12. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.13. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.14. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.15. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.16. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.17. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.18. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.19. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.

1.20. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards. The Arduino IDE is a software tool for creating and uploading code to Arduino boards.



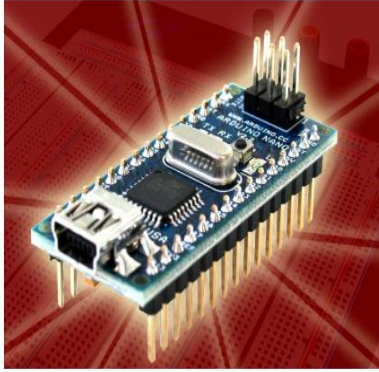
The Arduino company has joined the Impatto Zero policy of "Action is that each Arduino board produced is recycled, reused and not disposed of in landfills or incinerated."



Anexo D: Datasheet Arduino Nano

Arduino Nano (V2.3)

User Manual

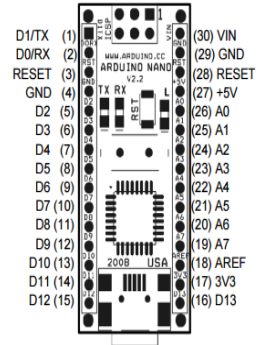


Released under the Creative Commons Attribution Share-Alike 2.5 License
<http://creativecommons.org/licenses/by-sa/2.5/>

More information:
www.arduino.cc

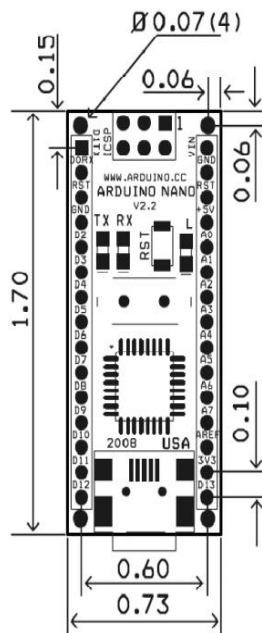
Rev. 2.3

Arduino Nano Pin Layout



Pin No.	Name	Type	Description
1-2, 5-16	D0-D13	I/O	Digital input/output port 0 to 13
3, 28	RESET	Input	Reset (active low)
4, 29	GND	PWR	Supply ground
17	3V3	Output	+3.3V output (from FTDI)
18	AREF	Input	ADC reference
19-26	A7-A0	Input	Analog input channel 0 to 7
27	+5V	Output or Input	+5V output (from on-board regulator) or +5V (input from external power supply)
30	VIN	PWR	Supply voltage

Arduino Nano Mechanical Drawing



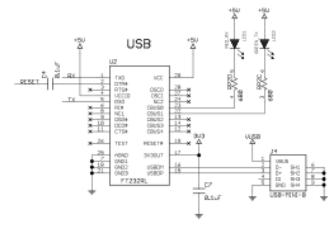
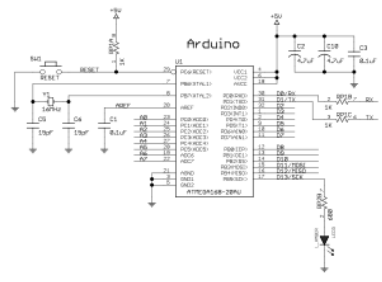
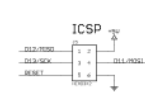
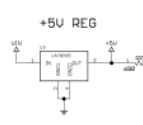
ALL DIMENTIONS ARE IN INCHES

Arduino Nano Bill of Material

Item Number	Qty	Ref. Desct.	Description	Mfg. P/N	MFG	Vendor P/N	Vendor
1	5	C1,C14,C17,C3	Capacitor, 0.1uF 50V 10% Ceramic 178 0805	C0805C104SRACU	Kemet	80-C0805C104SR	Mouser
2	3	C2,C3,C10	Capacitor, 4.7uF 10V 10% Tantalum Case A	T8914475K010AT	Kemet	80-T8914475K010	Mouser
3	2	C5,C5	Capacitor, 100pF 50V 5% Ceramic N0P/C06 0805	C0805C180J56ACTU	Kemet	80-C0805C180J56	Mouser
4	1	D1	Diode, Schottky 0.5A 20V	M8R0520J71G	OnSemi	863-M8R0520J71G	Mouser
5	1	H1,H2	Header, 50P5 1 Row	68000-126HLF	FCI	648-68000-126HLF	Mouser
6	1	J4	Connector, Mini-B Recept	67503-102D	Molex	538-67503-102D	Mouser
7	1	J5	Header, 70P5 2 Rows	67996-272HLF	FCI	648-67996-272HLF	Mouser
8	1	LD1	LED, Super Bright RED 100mcd 640nm 110degree 0805	APT2012SRCPV	Kingbright	604-APT2012SRCPV	Mouser
9	1	LD2	LED, Super Bright GREEN 50mcd 570nm 110degree 0805	APHM0012G6CK-F01	Kingbright	604-APHM0012G6CK	Mouser
10	1	LD3	LED, Super Bright ORANGE 150mcd 610nm 110degree 0805	APHM0012G6CK-F01	Kingbright	04-APHM0012G6CK	Mouser
11	1	LD4	LED, Super Bright BLUE 80mcd 470nm 110degree 0805	LTS1-C1078MT	Lite-On Inc	160-1579-1-ND	Digkey
12	1	R1	Resistor Pack, 1% +/-5% 62.5mW 48ES SMD	YC164-1R-0710L	Yageo	YC164-1-0KCT-ND	Digkey
13	1	R2	Resistor Pack, 600 +/-5% 62.5mW 48ES SMD	YC164-1R-07600RL	Yageo	YC164-1-600CT-ND	Digkey
14	1	SW1	Switch, Momentary Tact	SPST 150qF 3.0x2.5mm	Omron	SW100CT-ND	Digkey
15	1	U1	IC, Microcontroller RISC 16kB Flash, 0.5kB EEPROM, 23 I/O Pins	ATmega168-20AU	Atmel	556-ATMega168-20AU	Mouser
16	1	U2	IC, USB to SERIAL UART 28 Pins 520P	FT232RL	FTDI	895-FT232RL	Mouser
17	1	U3	IC, Voltage regulator 5V, 500mA 500P-223	UA78M50C0RG3	TI	595-UA78M50C0RG3	Mouser
18	1	Y1	Crystal, 16MHz +/-20ppm HC-49/US Low Profile	ABL-16100MHz-82	Abracorn	835-ABL-16-82	Mouser

Arduino Nano Schematic

Copyright 2008 under the Creative Commons Attribution Share-Alike 2.5 License
<http://creativecommons.org/licenses/by-sa/2.5/>



v2.3 - Modify FT232RL to use +5V	
TITLE:	Arduino Nano
Document Numbers:	BEU 2.3
Date:	6/26/2008 8:35:54 PM
Sheets:	L/1

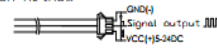
Anexo E: Sensor de Flujo de Agua Efecto Hall YF-S201

Flow-Pulse	Flow-Curve	Connection method
Flow Range: 100L/H-1800H-L/H		
Flow (L/H)	Frezaq.(HZ)	Erro range ± 10
120L/H	16	
240L/H	32.5	
360L/H	49.3	
480L/H	65.5	
600L/H	82	
720L/H	90.2	

Connection method:

YIFA the plastics Ltd Product Introduction

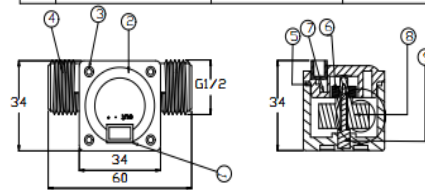
- 1.Mode:YF-21
- 2.Product Name:Hall sensor
- 3.Flow Range: 1-30L/MIN
- 4.(1)Connection Method




- (2)Voltage Range: 3.5-24VDC, Pulse Characteristic: $f=7Q(L/MIN)$
- (3)Extent of error: $\pm 5\%$
- (4)Flow-Pulse
2L/MIN=16HZ 4L/MIN=32.5HZ 6L/MIN=49.3HZ
8L/MIN=65.5HZ 10L/MIN=82HZ

5.Bom

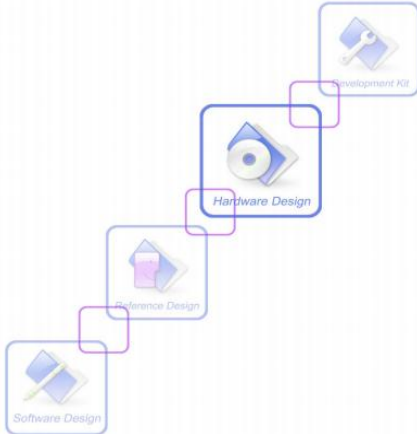
No.	Item	Material	Qty.
1	Connection wire		1
2	Bonnet	PA	1
3	Screw		4
4	Valve body	PA	1
5	Leak press valve		1
6	Magnet		1
7	Hall		1
8	Impeller	POM	1
9	Rustless steel axis	SUS304	1
10			
11			



Anexo F: Modulo de comunicación SIM800L



SIM800H_Hardware_Design_V1.00



Smart Machine Smart Decision

Document Title	SIM800H Hardware Design
Version	1.00
Date	2013-08-01
Status	Release
Document Control ID	SIM800H_Hardware_Design_V1.00

General Notes
SIMCom offers this information as a service to its customers, to support application and engineering efforts that use the products designed by SIMCom. The information provided is based upon requirements specifically provided to SIMCom by the customers. SIMCom has not undertaken any independent search for additional relevant information, including any information that may be in the customer's possession. Furthermore, system validation of this product designed by SIMCom within a larger electronic system remains the responsibility of the customer or the customer's system integrator. All specifications supplied herein are subject to change.

Copyright
This document contains proprietary technical information which is the property of SIMCom Limited, copying of this document and giving it to others and the using or communication of the contents thereof, are forbidden without express authority. Offenders are liable to the payment of damages. All rights reserved in the event of grant of a patent or the registration of a utility model or design. All specification supplied herein are subject to change without notice at any time.

Copyright © Shanghai SIMCom Wireless Solutions Ltd. 2013

Smart Machine Smart Decision

1. Introduction

This document describes SIM800H hardware interface in great detail. This document can help user to quickly understand SIM800H interface specifications, electrical and mechanical details. With the help of this document and other SIM800H application notes, user guide, users can use SIM800H to design various applications quickly.

2. SIM800H Overview

SIM800H is a quad-band GSM/GPRS module, that works on frequencies GSM850MHz, EGSM900MHz, DCS1800MHz and PCS1900MHz. SIM800H features GPRS multi-slot class 12/ class 10 (optional) and supports the GPRS coding schemes CS-1, CS-2, CS-3 and CS-4.

With a tiny configuration of 15.8*17.8*2.4mm, SIM800H can meet almost all the space requirements in user applications, such as smart phone, PDA and other mobile devices.

SIM800H has 88pin pads of LGA packaging, and provides all hardware interfaces between the module and customers' boards.

- Support 5*5*2 keypads
- One full modem serial port, user can configure two serial ports
- One USB, the USB interfaces can debug, download software
- Audio channel which includes two microphone input, a receiver output and a speaker output
- Programmable general purpose input and output.
- A SIM card interface
- Support Bluetooth
- Support FM
- Support one PWM

SIM800H is designed with power saving technique so that the current consumption is as low as 0.7mA in sleep mode.

2.1. SIM800H Key Features

Table 1: SIM800H key features

Feature	Implementation
Power supply	3.4V~4.4V
Power saving	typical power consumption in sleep mode is 0.7mA (BS-PA-MERMS#9)
Frequency bands	<ul style="list-style-type: none"> Quad-band: GSM 850, EGSM 900, DCS 1800, PCS 1900. SIM800H can search the 4 frequency bands automatically. The frequency bands can also be set by AT command "AT+CBAND?". For details, please refer to document [1]. Compliant to GSM Phase 22+
Transmitting power	<ul style="list-style-type: none"> Class 4 (2W) at GSM 850 and EGSM 900 Class 1 (1W) at DCS 1800 and PCS 1900
GPRS connectivity	<ul style="list-style-type: none"> GPRS multi-slot class 12 (default) GPRS multi-slot class 1-12 (option)

SIM800H_Hardware_Design_V1.00 11 2013-08-01

Smart Machine Smart Decision

Temperature range	<ul style="list-style-type: none"> Normal operation: -40°C ~ +85°C Storage temperature: -45°C ~ +90°C
Data GPRS	<ul style="list-style-type: none"> GPRS data downlink transfer: max. 85.6 kbps GPRS data uplink transfer: max. 85.6 kbps Coding scheme: CS-1, CS-2, CS-3 and CS-4 PAP protocol for PPP connect Integrate the TCP/IP protocol. Support Packet Broadcast Control Channel (PBCCH) CSD transmission rates: 2.4, 4.8, 9.6, 14.4 kbps
CSD	Support CSD transmission
USSD	Unstructured Supplementary Services Data (USSD) support
SMS	<ul style="list-style-type: none"> MT, MO, CB, Text and PDU mode SMS storage: SIM card
SIM interface	Support SIM card: 1.8V, 3V
External antenna	Antenna pad
Audio features	Speech codec modes: <ul style="list-style-type: none"> Half Rate (ETS 06.20) Full Rate (ETS 06.10) Enhanced Full Rate (ETS 06.50 / 06.60 / 06.80) Adaptive multi rate (AMR) Echo Cancellation Noise Suppression
Serial port and debug port	Serial port: <ul style="list-style-type: none"> Full modem interface with status and control lines, unbalanced, asynchronous. 1200bps to 115200bps. Can be used for AT commands or data stream. Support RTS/CTS hardware handshake and software ON/OFF flow control. Multiplex ability according to GSM 07.10 Multiplexer Protocol. Autobauding supports baud rate from 1200 bps to 57600bps. upgrading firmware Debug port: <ul style="list-style-type: none"> USB_DM and USB_DP Can be used for debugging and upgrading firmware.
Phonebook management	Support phonebook types: SM, FD, LD, RC, ON, MC.
SIM application toolkit	GSM 11.14 Release 99
Real time clock	Support RTC
Timing functions	Use AT command set
Physical characteristics	Size: 15.8*17.8*2.4mm Weight: 1.35g
Firmware upgrade	Main serial port or USB port.

Table 2: Coding schemes and maximum net data rates over air interface

Coding scheme	1 timeslot	2 timeslot	4 timeslot
CS-1	9.05kbps	18.1kbps	36.2kbps
CS-2	13.4kbps	26.8kbps	53.6kbps
CS-3	15.6kbps	31.2kbps	62.4kbps
CS-4	21.4kbps	42.8kbps	85.6kbps

2.2. Operating Mode

The table below summarizes the various operating modes of SIM800H.

Table 3: Overview of operating modes

Mode	Function	
Normal operation	GSM/GPRS SLEEP	Module will automatically go into sleep mode if the conditions of sleep mode are enabling and there is no on air and no hardware interrupt (such as GPIO interrupt or data on serial port). In this case, the current consumption of module will reduce to the minimal level. In sleep mode, the module can still receive paging message and SMS.
	GSM IDLE	Software is active. Module is registered to the GSM network, and the module is ready to communicate.
	GSM TALK	Connection between two subscribers is in progress. In this case, the power consumption depends on network settings such as DTX off/on, FR/EFRRH, hopping sequences, antenna.
	GPRS STANDBY	Module is ready for GPRS data transfer, but no data is currently sent or received. In this case, power consumption depends on network settings and GPRS configuration.
	GPRS DATA	There is GPRS data transfer (PPP or TCP or UDP) in progress. In this case, power consumption is related with network settings (e.g. power control level), uplink/downlink data rates and GPRS configuration (e.g. used multi-slot settings).
Power down	Normal power down by sending AT command "AT+CPWD=1" or using the PWRKEY. The power management unit shuts down the power supply for the baseband part of the module, and only the power supply for the RTC is remained. Software is not active. The serial port is not accessible. Power supply (connected to VBAT) remains applied.	
Minimum functionality mode	AT command "AT+CFUN" can be used to set the module to a minimum functionality mode without removing the power supply. In this mode, the RF part of the module will not work or the SIM card will not be accessible, or both RF part and SIM card will be closed, and the serial port is still accessible. The power consumption in this mode is lower than normal mode.	

2.3. Functional Diagram

The following figure shows a functional diagram of SIM800H:

- GSM baseband
- GSM RF
- Antenna interface
- Other interface

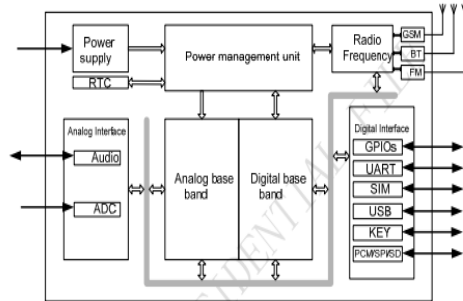


Figure 1: SIM800H functional diagram

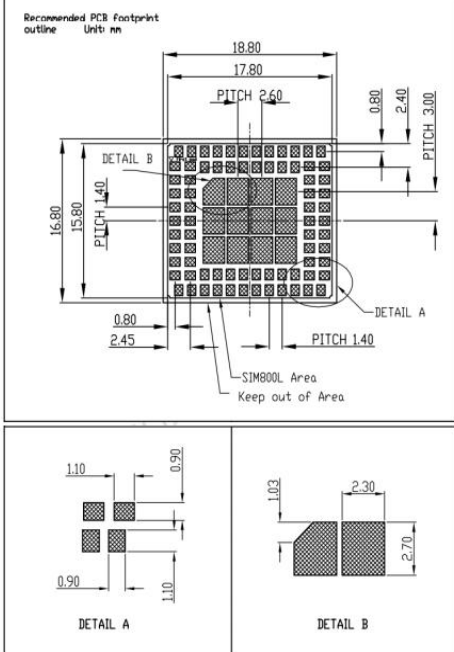


Figure 4: Recommended PCB footprint outline (Unit: mm)

4. Application Interface

4.1. Power Supply

The power supply range of SIM800H is from 3.4V to 4.4V. Recommended voltage is 4.0V. The transmitting burst will cause voltage drop and the power supply must be able to provide sufficient current up to 2A. For the VBAT input, a bypass capacitor (low ESR) such as a 100 μF is strongly recommended.

Increase the 33PF and 10PF capacitors can effectively eliminate the high frequency interference. A 5.1V/500mW Zener diode is strongly recommended, the diode can prevent chip from damaging by the voltage surge. These capacitors and Zener diode should be placed as close as possible to SIM800H VBAT pins.

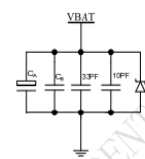


Figure 5: Reference circuit of the VBAT input

Table 5: Recommended Zener diode

Vendor	Part number	Power(watts)	Packages
1	On semi	MMSZ5231BTIG	500mW SOD123
2	Prsemi	PZ3D4V2H	500mW SOD323
3	Prsemi	PZ3D4V2H	500mW SOD523
4	Vishay	MMSZ4689-V	500mW SOD123
5	Crownco	CDZ55CSV1SM	500mW 0805

The following figure is the reference design of +5V input power supply. The designed output for the power supply is 4.1V, thus a linear regulator can be used.

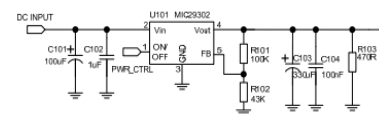


Figure 6: Reference circuit of the LDO power supply

7. Manufacturing

7.1. Top and Bottom View of SIM800H



Figure 54: Top and bottom view of SIM800H

7.2. Typical Solder Reflow Profile

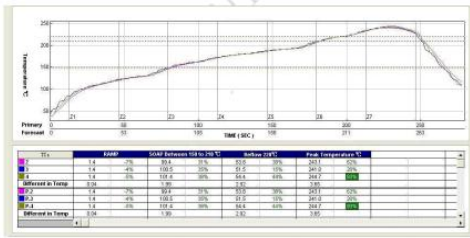


Figure 55: Typical solder reflow profile

7.3. The Moisture Sensitivity Level

The moisture sensitivity level of SIM800H is 3. The module should be mounted within 168 hours after unpacking in the environmental conditions of temperature 30°C and relative humidity of <math><50\%</math> (RH). It is

necessary to bake the module if the above conditions are not met:

Table 54: Moisture classification level and floor life

Level	Floor Life (out of bag) at factory ambient $\leq 30^{\circ}\text{C}/60\% \text{RH}$ or as stated
1	Unlimited at $\approx 30^{\circ}\text{C}/85\% \text{RH}$
2	1 year
2a	4 weeks
3	168 hours
4	72 hours
5	48 hours
5a	24 hours
6	Mandatory bake before use. After bake, it must be reflowed within the time limit specified on the label.

NOTES:

1. If the vacuum package is not open for 3 months or longer than the packing date, baking is also recommended before re-flow soldering.

2. For product handling, storage, processing, IPC / JEDEC J-STD-033 must be followed.

7.4. Baking Requirements

Because of its sensitivity to moisture absorption, SIM800H should be baked sufficiently before re-flow soldering. Otherwise SIM800H will be at the risk of permanent damage during re-flow soldering. SIM800H should be baked 192 hours at temperature $40^{\circ}\text{C}/5^{\circ}\text{C}/0^{\circ}\text{C}$ and <math><5\%</math> RH for low-temperature device containers, or 72 hours at temperature $80^{\circ}\text{C}/5^{\circ}\text{C}$ for high-temperature device containers. Care should be taken that the plastic tray is not heat resistant, SIM800H modules should be taken out for baking, otherwise the tray may be damaged by high-temperature during baking.

Table 55: Baking requirements

Baking temperature	Moisture	Time
$40^{\circ}\text{C}/5^{\circ}\text{C}$	<math><5\%</math>	192 hours
$120^{\circ}\text{C}/5^{\circ}\text{C}$	<math><5\%</math>	4 hours

8. Appendix

1. Related Documents

Table 56: Related documents

SN	Document name	Remark
[1]	SIM800 Series_AT Command Manual	
[2]	ITU-T Draft new recommendation V.25ter	Serial asynchronous automatic dialing and control
[3]	GSM 07.07	Digital cellular telecommunications (Phase 2+); AT command set for GSM Mobile Equipment (ME)
[4]	GSM 07.10	Support GSM 07.10 multiplexing protocol
[5]	GSM 07.05	Digital cellular telecommunications (Phase 2+); Use of Data Terminal Equipment – Data Circuit terminating Equipment (DTE – DCE) interface for Short Message Service (SMS) and Cell Broadcast Service (CBS)
[6]	GSM 11.14	Digital cellular telecommunications system (Phase 2+); Specification of the SIM Application Toolkit for the Subscriber Identity Module – Mobile Equipment (SIM – ME) interface
[7]	GSM 11.11	Digital cellular telecommunications system (Phase 2+); Specification of the Subscriber Identity Module – Mobile Equipment (SIM – ME) interface
[8]	GSM 03.38	Digital cellular telecommunications system (Phase 2+); Alphabets and language-specific information
[9]	GSM 11.10	Digital cellular telecommunications system (Phase 2); Mobile Station (MS) conformance specification; Part 1: Conformance specification
[10]	AN_Serial Port	AN_Serial Port

Anexo G: Código del módulo Recolección, Visualización y Control

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
#include <Ethernet.h>
#include <SoftwareSerial.h>
#include <String.h>
//Declaremos los pines CE y el CSN
#define CE_PIN 7
#define CSN_PIN 8
//Variable con la dirección del canal que se va a leer
const          uint64_t          N1[4]          =
{0xF0F0F0F0E1LL,0xA0A0A3A301,0xB0F0F0F0D3LL,0xC0F0F0F0D4LL};
const uint64_t N2[2] = {0xF0F0F0F0E3LL,0xB0F0F0F0D3LL};
const uint64_t N3[2] = {0xF0F0F0F0E4LL,0xC0F0F0F0D4LL};
//creamos el objeto radio (NRF24L01)
RF24 radio(CE_PIN, CSN_PIN);
int q=0;
int s=0;
int j=1;
//vector para los datos recibidos
float datosN1[7];
float datosN2[7];
float datosN3[7];
float datosto[7];
int asig[9];
//int asignod[9];
float promen1[8];
float promen2[8];
float promen3[8];
float promen11[8];
float promen21[8];
float promen31[8];
int cpro[8];
```

```
int encendido[5];
int pin = 30;
int pin1 =31;
int Pin2= 32;
int Pin3 =33;
int Pin4=34;
int Pin5=3;
int Pin6=4;
    int estado[5]={1,1,1,1,1};
float sen1[20];
float sen2[20];
float sen3[20];
float sen4[20];
float sen5[20];
float sen6[20];
//Declaración de la direcciones MAC e IP. También del puerto 80
byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED}; //MAC
byte ip[] = { 192,168,1,155 }; ; //IP
byte servery[] = { 192,168,1,150 };
EthernetServer servidor(80);
EthernetClient cliente1;
String readString = String(100);
boolean Pin3ON = false; // Status flag
boolean Pin4ON = false; // Status flag
boolean Pin5ON = false; // Status flag
boolean Pin6ON = false; // Status flag
boolean Pin7ON = false; // Status flag
boolean Pin8ON = false; // Status flag
boolean Pin9ON = false; // Status flag
boolean Pin10ON = false; // Status flag
boolean Pin11ON = false; // Status flag
boolean Pin12ON = false; // Status flag
boolean Pin13ON = false; // Status flag
boolean Pin14ON = false; // Status flag
boolean Pin15ON = false; // Status flag
boolean Pin16ON = false; // Status flag
```

```

boolean Pin17ON = false; // Status flag
void setup(){
//Abrimos el canal de Lectura
    radio.begin();
radio.openWritingPipe(N1[0]); // Abro el canal "0" para escribir
    // Establece el retardo y el número de reintentos tras fallo en la comunicación RF
    radio.setRetries(15,15);
Ethernet.begin(mac, ip); //Inicializamos con las direcciones asignadas
servidor.begin();
pinMode(Pin3, OUTPUT);
pinMode(Pin4, OUTPUT);
pinMode(Pin5, OUTPUT);
pinMode(Pin6, OUTPUT);
Serial.begin(9600);
}
//***** control *****/

void CONTROLdeRED()
{
int porcen1 = (promen11[1])*60/100;// el segundo sensor del ramal
int porcen2= (promen11[2]+promen31[1])*60/100;
int porcen3 = (promen21[1]+promen21[2])*60/100;
int porcen4 = (promen31[2])*60/100;
estado[4]=1;
if (Pin14ON==true){

    if(Pin7ON==true && Pin8ON==true)
    {
if ((promen21[1]<porcen1))
{
if((promen11[1]>2))
{
estado[0]=1;
    Pin10ON = false;
    Pin14ON = false;
}
}
}
}
}

```

```

else
{estado[0]=0; Pin10ON = true;}
}
else
{estado[0]=0; Pin10ON = true;}
}
else
{
    estado[0]=1;
    Pin10ON = false;
    Pin14ON = false;
}
}
if (Pin15ON==true)
{
if(Pin7ON==true && Pin9ON==true)
{
if (((promen31[2])< porcen2))
{
if((promen11[2]>2))
{
estado[1]=1;
Pin11ON = false;
Pin15ON = false;
CONSM();
}
else{ estado[1]=0; Pin11ON = true;}
}
else{ estado[1]=0; Pin11ON = true;}
}
}
else
{
    estado[1]=1;
    Pin11ON = false;
    Pin15ON = false;
}
}

```

```

}
if (Pin16ON==true){
if(Pin8ON==true && Pin9ON==true)
{
if ((promen31[1]< porcen3))
{
if(((promen21[1]+promen21[2])>2))
{
estado[2]=1;
Pin12ON = false;
Pin16ON = false;
CONSM();
}
else {estado[2]=0; Pin12ON = true;}
}
else {estado[2]=0; Pin12ON = true;}
}
else
{
estado[2]=1;
Pin12ON = false;
Pin16ON = false;
}
}
}
if (Pin17ON==true){
if(Pin8ON==true && Pin9ON==true)
{
if (promen21[2]< porcen4)
{
if(promen31[2]>2)
{
estado[3]=1;
Pin13ON = false;
Pin17ON = false;
CONSM();
}
}
}
}
}

```



```

    else {estado[3]=0; Pin13ON = true;}
}
else {estado[3]=0; Pin13ON = true;}
}
else
{
    estado[3]=1;
    Pin13ON = false;
    Pin17ON = false;
}
}
}
//*****CONTORLSMS
void CONSM(){
    digitalWrite(Pin3, HIGH);
}
//*****NODOS*****
void nodes()
{
    if(asi[5]==0)Pin7ON=false;
    else Pin7ON=true;
    if(asi[6]==0)Pin8ON=false;
    else Pin8ON=true;
    if(asi[7]==0)Pin9ON=false;
    else Pin9ON=true;
    if (Pin7ON==true && datosN1[4]==1) Pin3ON =true;
    else Pin3ON =false;
    if (Pin7ON==true && datosN1[5]==1) Pin4ON =true;
    else Pin4ON =false;
    if (Pin8ON==true && datosN2[4]==1) Pin5ON =true;
    else Pin5ON =false;
    if (Pin9ON==true && datosN3[4]==1) Pin6ON =true;
    else Pin6ON =false;
}
//*****promedios*****
void promedios ()

```

```

{
if (cpro[0]==0)cpro[0]=1;
    if (cpro[1]==0)cpro[1]=1;
        if (cpro[2]==0)cpro[2]=1;
        for(int j1 =0;j1 <= 6;j1++)
        {
        Serial.print(" ");
        Serial.print (datosN1[j1]);
        }
        Serial.println(".....");
        for(int j1 =0;j1 <= 6;j1++)
        {
        Serial.print(" ");
        Serial.print (datosN2[j1]);
        }
        Serial.println(".....");
        for(int j1 =0;j1 <= 6;j1++)
        {
        Serial.print(" ");
        Serial.print( datosN3[j1])
        }
        Serial.println(".....");
        Serial.println("ok11");
        promen11[1]=promen1[1]/cpro[0];
        promen11[2]=promen1[2]/cpro[0];
        promen21[1]=promen2[1]/cpro[1];
        promen21[2]=promen2[2]/cpro[1];
        promen31[1]=promen3[1]/cpro[2];
        promen31[2]=promen3[2]/cpro[2];
        CONTROLdeRED();
        }
void servidorr ()
{
if (cliente1.connect(servery, 80)>0) { // Conexion con el servidor
    cliente1.print("GET /base/tesi/uni.php?sen1="); // Enviamos los datos por GET
    cliente1.print(promen11[1]);
}
}

```

```
cliente1.print("&&");
cliente1.print("sen2=");
cliente1.print(promen11[2]);
cliente1.print("&&");
cliente1.print("sen3=");
cliente1.print(promen21[1]);
cliente1.print("&&");
cliente1.print("sen4=");
cliente1.print(promen21[2]);
cliente1.print("&&");
cliente1.print("sen5=");
cliente1.print(promen31[1]);
cliente1.print("&&");
cliente1.print("sen6=");
cliente1.print(promen31[2]);
cliente1.print("&&");
cliente1.print("node=");
cliente1.print(asig[0]);
cliente1.print("&&");
cliente1.print("actu1=");
cliente1.print(datosN1[4]);
cliente1.print("&&");
cliente1.print("actu2=");
cliente1.print(datosN1[5]);
cliente1.print("&&");
cliente1.print("actu3=");
cliente1.print(datosN2[4]);
cliente1.print("&&");
cliente1.print("actu4=");
cliente1.print(datosN3[4]);
cliente1.println();
Serial.println("Conectado");
} else {
    Serial.println("Fallo en la conexion");
}
if (!cliente1.connected()) {
```

```

    Serial.println("Disconnected!");
}
cliente1.stop();
cliente1.flush();
}
void loop(){
// Creacion De Coneccion Al Cliente
EthernetClient client = servidor.available();
if (client)
{
    while (client.connected()) { //Cliente conectado
char c = client.read();
        if (readString.length() < 100) //Leemos petición HTTP caracter a caracter
        {
            readString = readString + c;
        }
        Serial.print(c);
        if (c == '\n')
        {
            if(readString.indexOf("3=Habilitado") > -1)
            {
                Pin14ON = true;
            }
            if(readString.indexOf("4=Habilitado") > -1)
            {
                Pin15ON = true;
            }
            if(readString.indexOf("5=Habilitado") > -1)
            {
                Pin16ON = true;
            }
            if(readString.indexOf("6=Habilitado") > -1)
            {
                Pin17ON = true;
            }
        }
}
//-----HTML-----

```

```

client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println("Connection: close");
client.println("Refresh: 33");
client.println();
client.print("<html><head>");
client.print("<title>CONTROL DE FALLAS</title>");
client.println("</head>");
client.print("<body bgcolor='#C3DADE'>");
//---Titulo De La Pagina---
client.println("<br><hr />");
client.println("<h1><div align='center'><font color='#2076CD'>CONTROL DE FALLA PARA
UNA RED LOCAL DE AGUA </font color></div></h1>");
client.println("<hr /><br>");
//---Cabeza---
//---Cuerpo---
client.println("<table border='0' width=auto align='center' height=auto >");
client.println("<tr bgColor='#7DCEA0'>");
client.println("<td><font face='Verdana' color='#FFFFFF' size='5'>Sistema De Control De
Modulos</font></td>");
client.println("</tr></table>");
client.println("<table border='0' width='auto' cellpadding='9' cellspacing='5' align='center'
height=auto>");
client.println("<tr bgColor='#EAF1F1'><div style='text-align:center'>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>  SENSORES:
<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Sen 1<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Sen 2<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Sen 3<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Sen 4<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Sen 5<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Sen 6<br></font></td>");
client.println("</tr>");
client.println("<tr bgColor='#EAF1F1'>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>FLUJO VOLUMETRICO:
<br></font></td>");

```

```

//promedios();
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>");
client.print (promen11[1]);
client.print ("<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>");
client.print (promen11[2]);
client.print ("<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>");
client.print (promen21[1]);
client.print ("<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>");
client.print (promen21[2]);
client.print ("<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>");
client.print (promen31[1]);
client.print ("<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>");
client.print (promen31[2]);
client.print ("<br></font></td>");
client.println("</tr></table></div>");
client.println("<table border='0' width=auto align='center' height=auto >");
client.println("<tr bgColor='#7DCEA0'>");
client.println("<td><font face='Verdana' color='#FFFFFF' size='5'>VISUALIZACION DE
CONTROL</font></td>");
client.println("</tr></table>");
client.println("<table border='0' width=auto align='center'cellpadding='9'cellspacing='5'
height=auto >");
client.println("<tr bgColor='#7DCEA0'>");
client.println("<td><font face='Verdana' color='#FFFFFF' size='5'>NODO 1</font></td>");
client.println("<td><font face='Verdana' color='#FFFFFF' size='5'>NODO 2</font></td>");
client.println("<td><font face='Verdana' color='#FFFFFF' size='5'>NODO
3</font></td></tr>");
client.println("<tr bgColor='#7DCEA0'>");
if (Pin7ON)
client.println("<td align='center'><font color='green' size='5'>ON");
else

```

```

client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
if (Pin8ON)
client.println("<td align='center'><font color='green' size='5'>ON");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
if (Pin9ON)
client.println("<td align='center'><font color='green' size='5'>ON");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
client.println("</tr></table>");
client.println("<table border='0' width='auto' cellpadding='9' cellspacing='5' align='center'
height=auto>");
client.println("<tr bgColor='#EAF1F1'><div style='text-align:center'>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'> UBICACION:
<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF'
size='4'>VALVULA<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF'
size='4'>ESTADO<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF'
size='4'>CONTROL<br></font></td>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'><br></font></td>");
client.println("</tr>");
client.println("<tr bgColor='#EAF1F1'>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Ramal 1:
<br></font></td>");
if (Pin3ON)
client.println("<td align='center'><font color='green' size='5'>ON");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
if (Pin10ON)
client.println("<td align='center'><font color='green' size='5'>HABILITADO");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>DESABILITADO");
if (Pin14ON)
client.println("<td align='center'><font color='green' size='5'>ON");

```

```

else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
client.println("<td align='center' bgcolor='#EAF1F1'><form method=get><input type=submit
name=3 value='Habilitado'></form></td>");
client.println("</tr>");
client.println("<tr bgcolor='#EAF1F1'>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Ramal 2:
<br></font></td>");
if (Pin4ON)
client.println("<td align='center'><font color='green' size='5'>ON");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
if (Pin11ON)
client.println("<td align='center'><font color='green' size='5'>HABILITADO");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>DESABILITADO");
if (Pin15ON)
client.println("<td align='center'><font color='green' size='5'>ON");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
client.println("<td align='center' bgcolor='#EAF1F1'><form method=get><input type=submit
name=4 value='Habilitado'></form></td>");
client.println("</tr>");
client.println("<tr bgcolor='#EAF1F1'>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Ramal 3:
<br></font></td>");
if (Pin5ON)
client.println("<td align='center'><font color='green' size='5'>ON");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
if (Pin12ON)
client.println("<td align='center'><font color='green' size='5'>HABILITADO");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>DESABILITADO");
if (Pin16ON)
client.println("<td align='center'><font color='green' size='5'>ON");

```



```

else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
client.println("<td align='center' bgcolor='#EAF1F1'><form method=get><input type=submit
name=5 value='Habilitado'></form></td>");
client.println("</tr>");
client.println("<tr bgcolor='#EAF1F1'>");
client.println("<td><font face='Verdana' color='#CFCFCF' size='4'>Ramal 4:
<br></font></td>");
if (Pin6ON)
client.println("<td align='center'><font color='green' size='5'>ON");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
if (Pin13ON)
client.println("<td align='center'><font color='green' size='5'>HABILITADO");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>DESABILITADO");
if (Pin17ON)
client.println("<td align='center'><font color='green' size='5'>ON");
else
client.println("<td align='center'><font color='#B03A2E' size='5'>OFF");
client.println("<td align='center' bgcolor='#EAF1F1'><form method=get><input type=submit
name=6 value='Habilitado'></form></td>");
client.println("</tr></table></div>");
client.println("<table border='0' width='auto' cellpadding='9' cellspacing='5' align='center'
height=auto>");
client.println("<tr bgcolor='#EAF1F1'><div style='text-align:center'>");
client.println("<td align='center' bgcolor='#EAF1F1'><form method=get><input type=submit
name=7 value='confirmar'></form></td>");
client.println("</tr></table></div>");
readString=" ";
client.stop();
}
}}
radio.startListening();
radio.openReadingPipe(1,N1[j]);
radio.startListening();

```

```

if ( radio.available( ))
{
    //Leemos los datos y los guardamos en la variable datos[]
    radio.read(datosto,sizeof(datosto));
}
else
{
    for(int j1 =0;j1 <= 6;j1++)
    {
        datosto[j1]=0;
    }
}
s=s+1;
delay(1000);
radio.stopListening();
int k;
k=datosto[0];
switch (k)
{
    case 1:
    for(int j1 =0;j1 <= 6;j1++)
    {
        datosN1[j1]=datosto[j1];
    }
    promen1[1]=promen1[1]+datosN1[1];
    promen1[2]=promen1[2]+datosN1[2];
    cpro[0]=cpro[0]+1;
    asig[1]=1;
    sen1[cpro[0]]=datosN1[1];
    sen1[cpro[0]]=datosN1[2];
    break;
    case 2:
    for(int j1 =0;j1 <= 6;j1++)
    {
        datosN2[j1]=datosto[j1];
    }
}

```

```

    promen2[1]=promen2[1]+datosN2[1];
promen2[2]=promen2[2]+datosN2[2];
cpro[1]=cpro[1]+1;
asig[2]=1;
    sen3[cpro[0]]=datosN2[1];
    sen4[cpro[0]]=datosN2[2];
    break;
case 3:
for(int j1 =0;j1 <= 6;j1++)
{
    datosN3[j1]=datosto[j1];
}
promen3[1]=promen3[1]+datosN3[1];
promen3[2]=promen3[2]+datosN3[2];
cpro[2]=cpro[2]+1;
asig[3]=1;
    sen5[cpro[0]]=datosN3[1];
    sen6[cpro[0]]=datosN3[2];
j=1;
break;
}
j=j+1;
if(j>3)j=1;
delay(1000);
s=s+1;
radio.stopListening();
asig[0]=asig[1]*100+asig[2]*10+asig[3]*1;
    for(int j1 =0;j1 <= 3;j1++)
    {
Serial.print (datosN1[j1]);
Serial.print(" ");
    }
Serial.println(" ");
    for(int j1 =0;j1 <= 3;j1++)
    {
Serial.print (datosN2[j1]);

```

```

Serial.print("      ");

}
Serial.println("      ");
for(int j1 =0;j1 <= 3;j1++)
{
Serial.print (datosN3[j1]);
Serial.print("      ");
}
Serial.println("      ");
if (s>30)
{
asig[5]=asig[1];
asig[6]=asig[2];
asig[7]=asig[3];
nodes();
cpro[7]=cpro[7]+1;
if (asig[1]==0)
{
for(int j1 =0;j1 <= 5;j1++)
{
datosN1[j1]=0;
}
}
if (asig[2]==0)
{
for(int j1 =0;j1 <= 5;j1++)
{
datosN2[j1]=0;
}
}
if (asig[3]==0)
{
for(int j1 =0;j1 <= 5;j1++)
{
datosN3[j1] =0;
}
}
}

```

```

}
asig[1]=0;
asig[2]=0;
asig[3]=0;
s=0;
}
if(cpro[7]>=2){
promedios();
client.stop();
servidorr();
servidor.begin();
    for(int j1 =0;j1 <= 5;j1++)
    {
        datosN2[j1]=0;
    }
    for(int j1 =0;j1 <= 5;j1++)
    {
        datosN3[j1]=0;
    }
    for(int j1 =0;j1 <= 5;j1++)
    {
        datosN1[j1]=0;
    }
promen1[1]=0;
promen1[2]=0;
promen2[1]=0;
promen2[2]=0;
promen3[1]=0;
promen3[2]=0;
cpro[7]=0;
cpro[1]=0;
cpro[2]=0;
cpro[0]=0;
}
bool ok = radio.write(estado, sizeof(estado));
//reportamos por el puerto serial los datos enviados

```

```
if(ok)
{
bool ok = radio.write(estados, sizeof(estados));
}
s=s+1;
delay(1000);
Serial.print("-----");
Serial.println(s);
digitalWrite(Pin3, LOW);
}
```

Anexo H: Código del módulo Lectura, Transmisión y Activación

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
//Declaremos los pines CE y el CSN
#define CE_PIN 9
#define CSN_PIN 10
//Variable con la dirección del canal por donde se va a transmitir
const uint64_t canal[2] = {0xF0F0F0F0E1LL,0xA0A0A3A301};
//creamos el objeto radio (NRF24L01)
RF24 radio(CE_PIN, CSN_PIN);
//vector con los datos a enviar
float datos[7];
int estado_control[5];
    int frecuencia1[3];
    int ses=0;
int on2[1];
int on1[1];
int LED = 6; // Pin digital 6 para el LED
int LED1 = 7;
volatile int NumPulsos1[3];
volatile int NumPulsos[3]; // variable para cantidad de pulsos recibidos
int PinSensor = 2; //Sensor conectado en el pin 2
float factor_conversion1=7.34;
float factor_conversion2=7.20; //para convertir de frecuencia a caudal
int PinSen = 3; //Sensor conectado en el pin 2
//---Función que se ejecuta en interrupción-----
//incrementamos la variable de pulsos
void ContarPulsos ()
{
    NumPulsos[0]++;
    //delay(500);
```

```

}
void ContarPulsos1 ()
{
  NumPulsos1[1]++;
  //delay(500);
}
//---Funci3n para obtener frecuencia de los pulsos-----
void ObtenerFrecuencia ()
{
  frecuencia1[0]=0;
  frecuencia1[1]=0;
  NumPulsos[0] = 0;
  NumPulsos1[1] = 0; //Ponemos a 0 el n3mero de pulsos
  interrupts(); //Habilitamos las interrupciones
  delay(1000); //muestra de 1 segundo
  noInterrupts();
  frecuencia1[0]= NumPulsos[0] ;
  frecuencia1[1]= NumPulsos1[1] ;
}
int c=0;
void setup()
{
  //inicializamos el NRF24L01
  radio.begin();
  //inicializamos el puerto serie
  Serial.begin(9600);
  // Establece el retardo y el n3mero de reintentos tras fallo en la comunicaci3n RF
  radio.setRetries(15,15);
  pinMode(PinSensor, INPUT);
  attachInterrupt(digitalPinToInterrupt(PinSensor),ContarPulsos,RISING); //Interrupcion
0(Pin2),funcion,Flanco de subida)
  pinMode(PinSen, INPUT);
  attachInterrupt(digitalPinToInterrupt(PinSen),ContarPulsos1,RISING); //Interrupcion
0(Pin2),funcion,Flanco de subida)
  // Abro el canal "0" para leer
  radio.openWritingPipe(canal[1]); // Abro el canal "1" para escribir

```



```

digitalWrite(LED, LOW);
digitalWrite(LED1, LOW);
}
int y=0;
int yx=0;
void loop()
{
// ***** comunicacion al nodo *****
do{
ObtenerFrecuencia();
float caudal_L_m[5];
  Serial.println(frecuencia1[0]);
  Serial.println(frecuencia1[1]);
datos[4]=y;
datos[5]=yx;
float caudal_L_h[2];
  radio.stopListening(); // Paro de escuchar por el canal "1"
caudal_L_m[1]=frecuencia1[0]/factor_conversion1; //calculamos el caudal en L/m
caudal_L_m[2]=frecuencia1[1]/factor_conversion2; //calculamos el caudal en L/m
caudal_L_m[0]=1;
// caudal_L_m[1]=1.1;
//caudal_L_m[2]=1.2;
caudal_L_m[3]=0;
  Serial.println( caudal_L_m[1]);
  Serial.println( caudal_L_m[2]);
//cargamos los datos en la variable datos[]
datos[0]=caudal_L_m[0];
datos[1]=caudal_L_m[1];
datos[2]=caudal_L_m[2];
datos[3]=caudal_L_m[3];
bool ok = radio.write(datos, sizeof(datos));
//reportamos por el puerto serial los datos enviados
if(ok)
{
  ses=0;
}
}

```

```

else
{

    Serial.println("no se ha podido enviar");
    ses=ses+1;
}
c=c+1;
Serial.print("Datos enviados: S1...");
    Serial.print(datos[0]);
    Serial.print(" ,S2.... ");
    Serial.print(datos[1]);
    Serial.print(" , ");
    Serial.println(datos[2]);
Serial.println("-----");
}while(c!=4);
radio.openReadingPipe(1,canal[0]);
    radio.startListening(); // Comienzo a escuchar por el canal "0"
radio.read(estado_control, sizeof(estado_control));
if ( radio.available( )
{
    on1[0]=estado_control[0];
    if (on1[0] == 0)
    {
        digitalWrite(LED, HIGH);
        y=1;
    }
    if (on1[0] ==1)
    {
        digitalWrite(LED, LOW);
        y=0;
        Serial.println("55555555");
    }
    on2[0]=estado_control[1] ;
    if (on2[0]== 0)
    {
        digitalWrite(LED1, HIGH);

```

```
    yx=1;
  }

  if (on2[0]==1)
  {
    digitalWrite(LED1, LOW);
    yx=0;
  }
}
else
{
  Serial.println("No hay datos de radio disponibles");
}
c=0;
datos[4]=y;
  datos[5]=yx;
  delay(1000);
Serial.println(y);
  if (ses>=500)
  {
    ses=0;
    digitalWrite(LED, LOW);
    digitalWrite(LED1, LOW);
  }
}
```

Anexo I: Código de calibración del sensor

```
volatile int pulsos;
// Cantidad de pulsos del sensor. Como se usa dentro de una interrupción debe ser volatile
#define sensorDeFlujo 2
//Pin al que se conecta el sensor. Debe ser 2 porque es único que acepta interrupciones en el
Arduino UNO
void flujo ()
// Función de interrupción
{pulsos++; // Simplemente sumar el número de pulsos}
void setup()
{ pinMode(sensorDeFlujo, INPUT);
Serial.begin(9600);
attachInterrupt(0, flujo, RISING);
interrupts ();
}
void loop ()
{Serial.print("Pulsos: ");
Serial.println(pulsos);}
```

Anexo J: Tabla de correlación

Valor	Significado
-1	Correlación negativa grande y perfecta
-0,9 a -0,99	Correlación negativa muy alta
-0,7 a -0,89	Correlación negativa alta
-0,4 a -0,69	Correlación negativa moderada
-0,2 a -0,39	Correlación negativa baja
-0,01 a -0,19	Correlación negativa muy baja
0	Correlación nula
0,01 a 0,19	Correlación positiva muy baja
0,2 a 0,39	Correlación positiva baja
0,4 a 0,69	Correlación positiva moderada
0,7 a 0,89	Correlación positiva alta
0,9 a 0,99	Correlación positiva muy alta
1	Correlación positiva grande y perfecta

Anexo K: Código de configuración Arduino y la base de datos

```
<?php
    // config.php
    // Credenciales
    $dbhost = "localhost";
    $dbuser = "proyecto2";
    $dbpass = "123456";
    $dbname = "control";
    // Conexión con la base de datos
    $con = mysqli_connect($dbhost, $dbuser, $dbpass, $dbname);
?>
```

Anexo L: Código de enlace de datos Arduino y la base de datos

```
<?php
// iot.php
// Importamos la configuración
require("config.php");
// Leemos los valores que nos llegan por GET
$sensor1 = mysqli_real_escape_string($con, $_GET['sen1']);
$sensor2 = mysqli_real_escape_string($con, $_GET['sen2']);
$sensor3 = mysqli_real_escape_string($con, $_GET['sen3']);
$sensor4 = mysqli_real_escape_string($con, $_GET['sen4']);
$sensor5 = mysqli_real_escape_string($con, $_GET['sen5']);
$sensor6 = mysqli_real_escape_string($con, $_GET['sen6']);
$nodo = mysqli_real_escape_string($con, $_GET['node']);
$act1 = mysqli_real_escape_string($con, $_GET['actu1']);
$act2 = mysqli_real_escape_string($con, $_GET['actu2']);
$act3 = mysqli_real_escape_string($con, $_GET['actu3']);
$act4 = mysqli_real_escape_string($con, $_GET['actu4']);
$query = "INSERT INTO sensado(S1,S2,S3,S4,S5,S6,C_Nodo,A1,A2,A3,A4)
VALUES('".$sensor1."','".$sensor2."','".$sensor3."','".$sensor4."','".$sensor5."','".$sensor6."','".$nodo."','".$act1."','".$act2."','".$act3."','".$act4."')";
// Ejecutamos la instrucción
mysqli_query($con, $query);
mysqli_close($con);
?>
```

Anexo M: Fotografías del prototipo implementado



Anexo N: Página web implantada

CONTROL DE FALLA PARA UNA RED LOCAL DE AGUA

Sistema De Control De Modulos

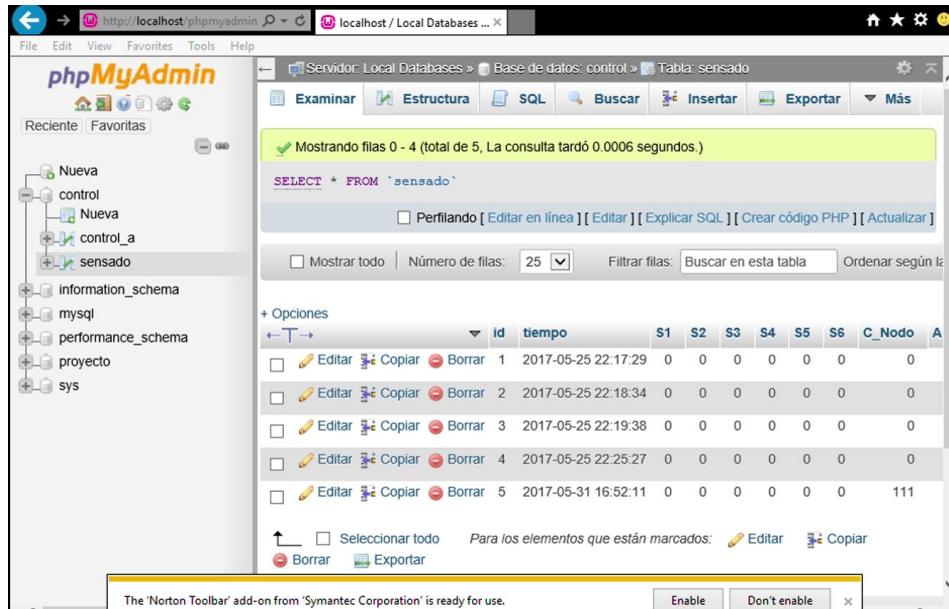
SENSORES:	Sen 1	Sen 2	Sen 3	Sen 4	Sen 5	Sen 6
FLUJO VOLUMETRICO:	2.12	1.65	1.19	0.00	0.00	0.49

VISUALIZACION DE CONTROL

NODO 1	NODO 2	NODO 3
ON	ON	ON

UBICACION:	VALVULA	ESTADO	CONTROL	
Ramal 1:	ON	HABILITADO	ON	<input type="button" value="Habilitado"/>
Ramal 2:	ON	HABILITADO	ON	<input type="button" value="Habilitado"/>
Ramal 3:	ON	HABILITADO	ON	<input type="button" value="Habilitado"/>
Ramal 4:	ON	HABILITADO	ON	<input type="button" value="Habilitado"/>

Anexo O: Fotografía de la base de datos implantada



The screenshot displays the phpMyAdmin interface for a local database. The main window shows the 'sensado' table with the following data:

	id	tiempo	S1	S2	S3	S4	S5	S6	C_Nodo	A
<input type="checkbox"/>	1	2017-05-25 22:17:29	0	0	0	0	0	0	0	0
<input type="checkbox"/>	2	2017-05-25 22:18:34	0	0	0	0	0	0	0	0
<input type="checkbox"/>	3	2017-05-25 22:19:38	0	0	0	0	0	0	0	0
<input type="checkbox"/>	4	2017-05-25 22:25:27	0	0	0	0	0	0	0	0
<input type="checkbox"/>	5	2017-05-31 16:52:11	0	0	0	0	0	0	111	

At the bottom of the screenshot, a notification box states: "The 'Norton Toolbar' add-on from 'Symantec Corporation' is ready for use." with "Enable" and "Don't enable" buttons.