



**UNIVERSIDAD DEL BÍO-BÍO**  
FACULTAD DE INGENIERÍA  
DEPTO. INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

# “DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE CONTROL DOMÓTICO BASADO EN EL MICROCONTROLADOR ATMEGA 2560 PARA DISPOSITIVOS MÓVILES”

*Autores:*

Michael Eduardo Álvarez Ferrada

Diego Alexis Grandón Campos

SEMINARIO PARA OPTAR AL TÍTULO DE  
INGENIERO DE EJECUCIÓN EN ELECTRÓNICA

CONCEPCIÓN – CHILE

2018



**UNIVERSIDAD DEL BÍO-BÍO**  
FACULTAD DE INGENIERÍA  
DEPTO. INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

# “DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE CONTROL DOMÓTICO BASADO EN EL MICROCONTROLADOR ATMEGA 2560 PARA DISPOSITIVOS MÓVILES”

*Autores:*

Michael Eduardo Álvarez Ferrada

Diego Alexis Grandón Campos

*Docente Patrocinante:*

John Correa Toloza

*Docentes Adjuntos o Correctores*

Gustavo Sanhueza Garrido

Pablo Sáez Srain

## Tabla de Contenidos

Resumen .....	6
Introducción.....	7
Capítulo 1. Introducción a la Domótica.....	8
1.1 ¿Qué es Domótica? .....	8
1.2 Vivienda Domótica .....	8
1.3 Sistema Domótico.....	9
Capítulo 2 . Hardware de desarrollo.....	10
2.1 Plataformas de desarrollo.....	10
2.2 Plataformas open and closed.....	10
2.3 Open hardware .....	11
2.4 Tarjetas de desarrollo.....	12
2.4.1 Raspberry PI 3.....	12
2.4.2 beaglebone black.....	13
2.4.3 Pic Pingüino 18F4550 .....	14
Capítulo 3 . Plataforma Arduino.....	15
3.1 ¿Qué es Arduino?.....	15
3.2 ¿Cuál es el origen de Arduino?.....	17
3.3 Tarjetas Arduino .....	18
3.4 Arduino mega 2560 .....	19
3.5 Software Arduino.....	21
3.5.1 Programación Arduino .....	22
Capítulo 4 . Comunicación Bluetooth .....	32
4.1 Comunicación Bluetooth y Arduino .....	32
4.2 Conexión Bluetooth sistema Android.....	37
4.3 Arquitectura aplicación App Inventor .....	38
4.3.1 Componentes.....	39

4.3.2	Comportamiento.....	39
4.3.3	Eventos.....	40
4.4	Entorno de desarrollo.....	41
4.4.1	Diseñador de componentes.....	41
4.4.2	Editor de bloques.....	42
Capítulo 5 . Control Domótico .....		43
5.1.	Domótica .....	43
5.2.	Estructuración de un control domótico.....	44
5.3.	Transferencia de datos .....	45
5.4.	Arquitectura del sistema domótico .....	45
5.4.1.	Arquitectura con foco central .....	46
5.4.2.	Arquitectura Distribuida.....	47
5.4.3.	Arquitectura sin foco central .....	48
5.4.4.	Arquitectura Mixta .....	49
Capítulo 6 . Implementación .....		50
6.1.	Primera Fase .....	50
6.2.	Arduino y Actuadores y Sensores .....	51
6.3.	Plano de Maqueta .....	52
6.4.	Plano Eléctrico.....	53
6.5.	Interfaz Aplicación Móvil .....	54
Capítulo 7 . Ejecución del Sistema.....		55
7.1.	Conexión Físico .....	55
7.2.	Implementación Maqueta .....	56
7.3.	Sistema de Luces .....	57
7.4.	Verificación de la Comunicación .....	58
7.5.	Sistema de Alarmas .....	59
7.6.	Entradas On/Off.....	60

7.7 Ventilación y Temperatura .....	61
Conclusiones y Comentarios .....	63
Bibliografías .....	65
Anexos .....	66

## Resumen

En el presente proyecto de título se planea, diseña y confecciona un sistema domótico con comunicación de acceso vía bluetooth con el fin de lograr el control total de una vivienda, ya sea controlando las variables de encendido/apagado de luces de habitaciones, aire acondicionado, puertas principales, alarma anti robo y medición de temperatura, todo esto con un sistema de sencilla instalación y a un precio razonable.

Entrando en más detalle, el sistema domótico permite medir variables análogas del ambiente como temperatura y además permite controlar variables digitales, que representan los elementos físicos de la maqueta.

Para la confección del prototipo se utilizaron materiales de maqueta simple, en el cual, el control es realizado con por una placa electrónica comercial de la marca ARDUINO. Esta placa es programada en lenguaje C a través de la IDE Arduino, que se obtiene mediante la página web de la marca. Para lograr la comunicación Bluetooth se utiliza un Módulo Bluetooth HC-05 con conexión usb. Utilizando librerías en el IDE más la información recopilada en la web se logra hacer tangible un proyecto que hasta ahora eran solo diseños y planos de AutoCAD. Esto facilita la exposición de la domótica a niveles de entendimiento básicos, ya que involucra conocimientos relacionados al área de la electrónica e informática, lo que genera como resultado final una vivienda automatizada.

## **Introducción**

Mientras el tiempo avanza, el mundo evoluciona, por lo tanto, el ser humano tiene que adaptarse a los cambios. En este tiempo moderno la tecnología cambia de formas abruptas, lo que impacta en la cultura de la humanidad. Hoy en día, se busca comodidad y facilidad en las actividades diarias, además de seguridad. Las personas actuales prefieren tener el control total de sus vidas.

Debido a estas necesidades es que surge la idea de este proyecto, más que solucionar o cubrir una necesidad, trata de nivelar los estándares de conocimientos necesarios para la implementación de la misma, lograr que las personas tengan una base en el área de la electrónica como punta inicial. Por eso nuestro seminario es catalogado con finalidades académicas.

## Capítulo 1. Introducción a la Domótica

### 1.1 ¿Qué es Domótica?

Desde el punto de vista etimológico, la palabra domótica fue inventada en Francia y está formada por la contracción de “domus” (vivienda) más automática. La domótica es un término empleado en el área de la tecnología, para referirse a todo aquello que constituye el dominio y la supervisión de todos los elementos que integran una edificación compuesta por oficinas o que se encuentran sencillamente en una vivienda. Es un grupo de tecnologías que se encuentran adaptadas para ejercer el control y sistematización dentro de una vivienda, con la finalidad de poder proporcionar un eficiente uso de la energía, así como también aportar seguridad y comodidad al usuario.

Se pueden distinguir tres sectores distintos dependiendo del alcance de aplicación de esta tecnología:

- Domótica, para el sector doméstico.
- Inmótica, para el sector terciario e industrial (residencias, hoteles, zonas comunitarias, etc.)
- Urbótica, para las ciudades (control de la iluminación pública, gestión de semáforos, telecomunicaciones, medios de pago, etc.)

### 1.2 Vivienda Domótica

Para definir una vivienda domótica hay que tener en cuenta al menos dos puntos de vista: el del usuario y el punto de vista técnico. Desde el punto de vista del usuario una vivienda domótica es aquella que proporciona una mayor calidad de vida a través de la tecnología, desde el punto de vista técnico es aquella en la que se integran distintos dispositivos electrónicos que tienen la capacidad de intercomunicarse entre ellos a través de un soporte o hardware previamente programados.



### **1.3 Sistema Domótico**

Un sistema domótico es un conjunto de sensores, circuitos de procesamiento lógico y control, actuadores y fuente de alimentación.

Los sensores obtienen información del mundo físico externo y la transforman en una señal eléctrica que puede ser manipulada por la circuitería de control o por medio de un conjunto lógico de instrucciones. Existen sensores de todo tipo como por ejemplo de temperatura, humedad, movimiento y sonido, etc.

Los circuitos internos de un sistema domótico procesan las señales obtenidas de los sensores, la manipulación de dichas señales dependerá del diseño del hardware del sistema, como también del conjunto lógico de instrucciones que tenga pregrabado el hardware utilizado.

Los actuadores transforman la señal eléctrica acabada de procesar por el hardware en energía que interactúa con el mundo externo, ejemplo de actuadores son: un motor, ampollas, altavoces.

Y por último y no menos importante es la fuente de alimentación que proporciona la energía necesaria para poner en funcionamiento todo lo descrito anteriormente, algunos ejemplos de fuentes son: pilas, baterías, adaptadores AC/DC.

## **Capítulo 2 . Hardware de desarrollo**

### **2.1 Plataformas de desarrollo**

Las plataformas de desarrollo corresponden a tarjetas que son ensambladas en fábricas y son comercializadas o aquellas que sus placas de circuito impreso (PCB) que se encuentran a libre disposición para ser armados en casa. Una de las grandes ventajas de las plataformas de desarrollo es que existen comúnmente comunidades que entregan soporte y ayuda, ofreciendo un gran respaldo a la hora de efectuar un proyecto.

### **2.2 Plataformas open and closed**

En el desarrollo de proyectos electrónicos se encuentran distintos tipos de hardware y plataformas de desarrollo, pueden ser de tipo closed u open en el ámbito de hardware y software.

Las plataformas open se definen como aquellos sistemas que sus PCB, diagramas, dispositivos y software son de acceso público, donde los usuarios pueden realizar cambios a la estructura y realizar análisis de forma independiente sin necesidad de consultar al fabricante.

Por otro lado, las plataformas closed se encuentran con la limitación de información que solo entrega el fabricante, es por eso que estas plataformas de desarrollo se encuentran de forma comercial, sin la posibilidad de ser armadas por el usuario.

Es por esto, que al momento de desarrollar un proyecto se debe tener en cuenta la plataforma a seleccionar en el ámbito de la información que se encuentra disponible y la factibilidad que se le entrega a ingenieros, técnicos o usuarios casuales. Si se comparan los dos tipos de plataformas las dos poseen ventajas y desventajas, que nos ayuda a guiarnos en la elección del tipo de sistema más adecuado para un proyecto de control. Se puede afirmar que ninguna plataforma es mejor que la otra, ya que la mejor es la que más se adecua al proyecto que se desea realizar.

## 2.3 Open hardware

Generalmente el open hardware viene ligado directamente con el open software, la definición que se le atribuye es para artefactos, máquinas, dispositivos y sistemas que respetan la libertad de sus creadores de controlar su tecnología y al mismo tiempo compartir conocimiento y fomentar el comercio a través del intercambio abierto de diseños.

Sin embargo, para que un hardware se pueda denominar abierto es necesario que cumpla con los siguientes requisitos:

- Publicar la documentación incluyendo los archivos de los diseños, mismos que deben permitir su modificación y distribución.
- Ofrecer el software necesario para leer los archivos de diseño o la documentación suficiente de las funcionalidades requeridas, para que se pueda escribir el código de fuente abierta del mismo.
- Ofrecer una licencia que permita producir derivados y modificaciones, además de su redistribución bajo la licencia original, así como su venta y manufactura.
- La licencia no debe discriminar a ningún tipo de grupo o persona.
- La licencia no debe restringir a ningún campo o actividad. Es decir, no se puede limitar su uso únicamente para negocios o prohibir su uso para investigación nuclear.
- La licencia debe permitir el uso de otros componentes de hardware o software externos.
- La licencia tiene que ser neutral, ninguna disposición de la misma debe basarse en una tecnología específica, parte o componente, material o interfaz para su uso.

## 2.4 Tarjetas de desarrollo

En el comercio se pueden encontrar diversas tarjetas de desarrollo, ya sea como una plataforma open o closed, que poseen distintas características que se acomodan a los tipos de soluciones que se pueden desarrollar para un proyecto de complejidad alta o simple.

Al momento de desarrollar una solución para un proyecto muchas veces es más eficiente familiarizarse con una tarjeta de desarrollo, que comenzar un trabajo desde cero.

Las tarjetas de desarrollo permiten realizar prototipos para realizar ensayos lo que se hace muy cómodo para realizar pruebas. Algunos ejemplos de tarjetas de desarrollo son:

### 2.4.1 Raspberry PI 3

Raspberry es una computadora de placa simple o también conocida por las siglas SBC (Single Board Computer), esto significa que dichas placas son sistemas embebidos en las que se integran todos, o gran mayoría de los elementos que componen a un computador funcional.

Básicamente, Raspberry Pi es una mini computadora al que se le pueden conectar los mismos periféricos que a cualquier otro ordenador, tales como pantalla, teclado, ratón, pendrive, impresora, parlantes, etc.

Posee puertos USB, puerto LAN, salida de audio para Jack de 3.5mm, puerto para cámara, HDMI y ranura para micro SD y pines de conexión de entrada y salida para sensores y actuadores. El precio en el mercado es de 53,23 dólares.



Fig. 2.1. Raspberry PI 3

## 2.4.2 beaglebone black

Beaglebone Black es un computador de bolsillo de bajo costo. Incorpora un procesador Sitara AM3358 ARM Cortex-A8, producido por Texas Instruments, y una memoria SDRAM de 512Mb DDR3L. También posee sistema operativo Linux (distribución Linux Angstrom), precargado en la memoria Flash (MMC embebida de 4GB) y que se puede utilizar de forma inmediata, aun así, Beaglebone Black puede soportar distintos tipos de sistema operativo basados en Linux, tales como Ubuntu, Android y Fedora desde una memoria externa de tipo MicroSD.

Beaglebone Black también posee una salida HDMI tipo D para conectar una pantalla LCD o TV, por medio de la cual tendrás una interfaz gráfica de las aplicaciones. También incluye conexión a Ethernet de 10/100mbps, puerto USB 2.0, botón de reset, de encendido y 5 indicadores LEDs de estados (10/100mbps en Ethernet, alimentación, etc).

El precio en el comercio de beaglebone black 89,12 dólares

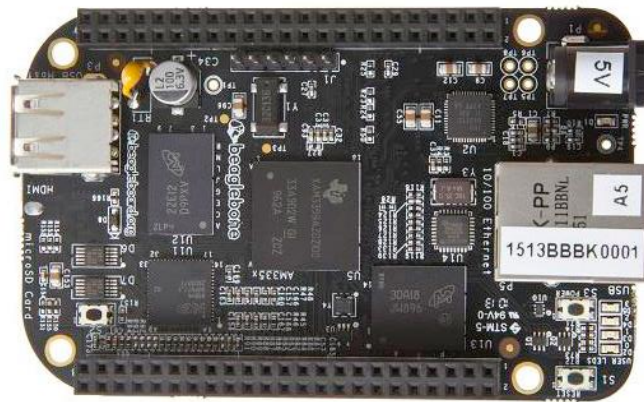


Fig. 2.2. Beaglebone black.

### 2.4.3 Pic Pingüino 18F4550

La tarjeta pingüino está construida con un chip de Microchip PIC18F4550. Funciona con un cristal de 20 MHz y es compatible con USB 2.0, las características de esta placa son corresponden a un microcontrolador con modulo USB 2.0 soporta Low speed 1.5Mb/s y full speed 12Mb/s. Posee 1KB de memoria de doble acceso vía USB, 29 pines I/O disponibles para la conexión de sensores y actuadores, una memoria flash de 32 KB, memoria RAM de 2048 Bytes, EEPROM de datos 256 Bytes, convertidor análogo digital de 10 bits y 8 canales. Esta tarjeta posee una tecnología nanowatt que brinda características y funcionamiento de bajo consumo y ahorro de energía, su voltaje de operación es de 4.2V a 5.5V.

Soporta 100 mil ciclos de borrado/escritura en la memoria flash, 1 millón de ciclos de borrado/escritura en la memoria EEPROM.

Su precio en el comercio es de 16,81 dólares.



Fig. 2.3. Pic pingüino 18F4550.

## Capítulo 3 . Plataforma Arduino

### 3.1 ¿Qué es Arduino?

Arduino es una placa de hardware libre que incorpora un microcontrolador reprogramable y una serie de pines-hembra (los cuales están unidos internamente a las patillas de I/O del microcontrolador) que permiten conectar de forma muy sencilla y cómoda diferentes sensores y actuadores.

Cuando hablamos de “placa hardware” nos estamos refiriendo en concreto a una PCB (placa de circuito impreso). Las PCBs son superficies fabricadas en un material no conductor sobre las cuales aparecen laminadas pistas de material conductor. Las PCBs se utilizan para conectar eléctricamente, a través de las pistas conductoras, diferentes componentes electrónicos soldados a ella. Una PCB es la forma más compacta y estable de construir un circuito electrónico. Entonces la placa Arduino no es más que una PCB que implementa un determinado diseño de circuitería interna.

No obstante, cuando hablamos de “placa Arduino”, deberíamos especificar el modelo concreto, ya que existen varias placas Arduino oficiales, cada una con diferentes características (como tamaño físico, el número de pines-hembra ofrecidos, el modelo de microcontrolador incorporado). Conviene conocer estas características para identificar que placa Arduino es la que nos convendrá más en cada proyecto.

De todas formas, aunque puedan ser modelos específicos diferentes, los microcontroladores incorporados en las diferentes placas Arduino pertenecen todos a la misma familia tecnológica, por lo que su funcionamiento en realidad es bastante parecido entre sí. En concreto, todos los microcontroladores son de tipo AVR, una arquitectura de microcontroladores desarrollada por la marca ATMEL.

El diseño del hardware de la placa Arduino está inspirado originalmente en el de otra placa de hardware libre preexistente, la placa WIRING. Esta placa surgió en el 2003 como proyecto personal de Hernando Barragán, estudiante del instituto de diseño de Ivrea.

Un software gratis, libre y multiplataforma (ya que funciona en Linux, MacOS y Windows) que debemos instalar en nuestro ordenador y que nos permite escribir, verificar y guardar en la memoria del microcontrolador de la placa Arduino el conjunto de instrucciones que deseamos que se empiece a ejecutar. Es decir que nos permite programarlo. La manera estándar de conectar nuestro computador con la placa Arduino para poder enviarle y grabarle dichas instrucciones es mediante un cable USB, gracias a que la mayoría de las placas Arduino incorporan un conector de este tipo.

Los proyectos Arduino pueden ser autónomos o no. En el primer caso, una vez programado su microcontrolador, la placa no necesita estar conectada a ningún computador y puede funcionar autónomamente si dispone de alguna fuente de alimentación. En el segundo caso, la placa debe estar conectada de alguna forma permanente (por cable USB, por cable de red Ethernet, etc.) a un computador ejecutando algún software específico que permita la comunicación entre este y la placa y el intercambio de datos entre ambos dispositivos. Este software específico lo debemos programar generalmente nosotros mediante algún lenguaje de programación estándar como Python, C, Java, Php. Y será independiente completamente del entorno de desarrollo Arduino, el cual no se necesitará más, una vez que la placa ya haya sido programada y esté en funcionamiento.

Tanto el entorno de desarrollo como el lenguaje de programación Arduino están inspirado en otro entorno y lenguaje libre preexistente: Processing, desarrollado inicialmente por Ben Fry y Casey Reas. Que el software Arduino se parezca tanto a Processing no es casualidad, ya que este está especializado en facilitar la generación de imágenes en tiempo real, de animaciones y de interacciones visuales, por lo que muchos profesores del instituto de Diseño de Ivrea lo utilizaban en sus clases. Como fue en ese centro precisamente se inventó Arduino es natural que ambos entornos y lenguajes guarden bastantes similitudes. No obstante, hay que aclarar que el lenguaje Processing es construido internamente con código escrito en lenguaje Java, mientras que el lenguaje Arduino se basa internamente en código C/C++.



### **3.2 ¿Cuál es el origen de Arduino?**

Arduino nació en el año 2005 en el instituto de diseño interactivo de Ivrea (Italia), centro académico donde los estudiantes se dedicaban a experimentar con la interacción entre humanos y diferentes dispositivos basados en microcontroladores, para conseguir generar espacios únicos, especialmente artísticos. Arduino apareció por la necesidad de contar con un dispositivo para utilizar en las aulas que fuera de bajo coste, que funcionase bajo cualquier sistema operativo y que contase con documentación adaptada a gente que quisiera empezar de cero. La idea original fue fabricar la placa para el uso interno de la escuela.

Sin embargo, el instituto se vio obligado a cerrar sus puertas precisamente en 2005. Ante la perspectiva de perder en el olvido todo el desarrollo del proyecto Arduino que se había ido llevando a cabo durante aquel tiempo, se decidió liberarlo y abrirlo a la comunidad para que todo el mundo tuviera la posibilidad de participar en la evolución del proyecto, proponer mejoras y sugerencias y mantenerlo vivo.

El principal responsable de la idea y diseño de Arduino, y la cabeza visible del proyecto es el llamado “Arduino Team”, formado por Massimo Banzi, David Cuartielles, David Mellis, Tom Igoe y Gianluca Martino.

### 3.3 Tarjetas Arduino

En el mercado existen distintas posibilidades de placas Arduino, estas variantes están especializadas en trabajar dentro de circunstancias específicas donde la placa ofrezca las soluciones necesarias que se necesitan para elaborar un proyecto.

Los tipos de placa que se pueden encontrar oficialmente en el comercio son:

- Arduino Uno
- Arduino Mega 2560
- Arduino Mega ADK
- Arduino Ethernet
- Arduino Fio
- Arduino Pro
- Arduino Lilypad
- Arduino Nano
- Arduino Mini
- Arduino Pro mini
- Arduino Leonardo
- Arduino Micro
- Arduino Due

La plataforma Arduino también incorpora tarjetas de tipo expansión que aumentan las capacidades iniciales de las placas. En esta categoría se pueden encontrar:

- Wifi Shield
- Ethernet Shield
- Bluetooth Shield
- Motor Shield

Si se desea elaborar un proyecto que tenga una conexión inalámbrica entre los distintos dispositivos que lo conforman, sería ideal utilizar la placa de expansión Wifi Shield o la Bluetooth Shield.

### 3.4 Arduino mega 2560

El Arduino mega 2560 (Fig. 3.1) está diseñado para proyectos más complejos, cuenta con 54 pines I/O digitales, 16 entradas analógicas, esta placa es recomendada para impresoras 3D, proyectos de automatización y robótica. Esta placa da a los proyectos mucho más espacio y oportunidades.



Fig. 3.1. Arduino Mega 2560.

Arduino Mega 2560 está basado en el microcontrolador ATmega2560, tiene una velocidad de reloj de 16MHz, una memoria flash de 256KB, SRAM de 8KB y 4KB de EEPROM.

La Arduino Mega 2560 (Fig. 3.2) cuenta con 16 entradas analógicas y 54 entradas digitales de las cuales 15 funcionan como salidas PWM de 8 bits. Posee un convertor análogo/ digital de 10 bits.

Se alimenta con una tensión externa de 7-12 VCD, la alimentación es un factor relevante al momento de conectar sensores o mantener en buen estado la placa, si se alimenta con menos de 7 VDC puede que los sensores no reciban los 5VDC necesarios para realizar una correcta medición. Al alimentar sobre los 12 VDC el regulador incorporado en la tarjeta se sobrecalienta, lo que con lleva a un daño en la placa.



Fig. 3.2. Elementos de la tarjeta Arduino Mega 2560.

Si se comparan las tarjetas mostradas anteriormente con la placa Arduino Mega 2560, las plataformas Raspberry Pi 3 y beaglebone black son ampliamente superiores en la velocidad de procesamiento, prácticamente son computadores de bolsillo, ya que estas plataformas se enfocan más en el desarrollo de softwares. Por consiguiente, Comparando con la tarjeta Pic Pingüino 18F4550, Arduino Mega 2560 supera sus características en el ámbito de su memoria RAM, memoria interna y EEPROM.

### 3.5 Software Arduino

Un programa es un conjunto concreto de instrucciones, ordenadas y agrupadas de forma adecuada y son ambigüedades que pretende obtener un resultado determinado. Cuando se dice que un microcontrolador es programable, estamos diciendo que permite grabar en su memoria de forma permanente el programa que deseemos que dicho microcontrolador ejecute. Si no introducimos ningún programa en la memoria del microcontrolador, este no sabrá que hacer.

Para poder desarrollar un programa nos hace falta un entorno de desarrollo, a lo que se le llama IDE. La sigla IDE viene de Integrated Development Environment, lo que traducido a nuestro idioma significa entorno de desarrollo integrado. Es simplemente una forma de llamar al conjunto de herramientas software que permite a los programadores poder desarrollar sus propios programas con comodidad. En el caso de Arduino, necesitamos un IDE que nos permita escribir y editar nuestro programa (llamado “Sketch” en el universo Arduino), que nos permita comprobar que nos hayamos cometido ningún error y que además nos permita, cuando estemos seguros de que el sketch es correcto, grabarlo en la memoria del microcontrolador de la placa Arduino para que este se convierta a partir de entonces en el ejecutor autónomo de dicho programa. En la Fig. 3.3 se muestra el IDE de Arduino.

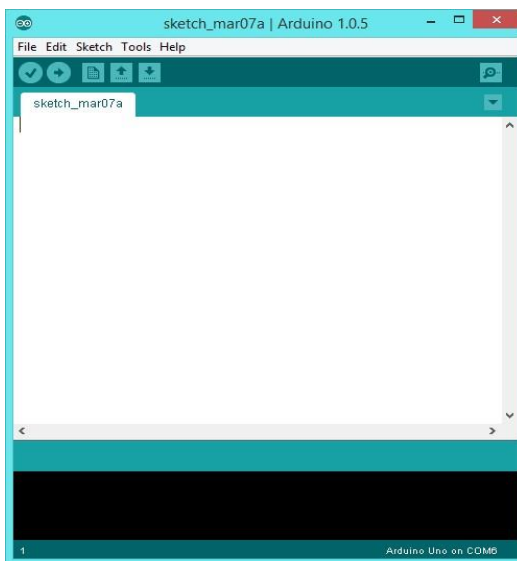


Fig. 3.3. IDE de Arduino.

### 3.5.1 Programación Arduino

Arduino se programa no solamente en un Sketch o en C, sino que ya hay una serie de lenguajes de programación que pueden usarse en esta plataforma. como, por ejemplo:

- **ArduBlock:** Construido para aquellos que empiezan a programar. En lugar de escribir código aquí se construyen los programas de forma visual usando bloques que contienen las instrucciones. Es lo equivalente a Scratch, un lenguaje de bloques creado en el MIT (Instituto Tecnológico de Massachusetts) para enseñar a los niños a programar (Fig.3.4).

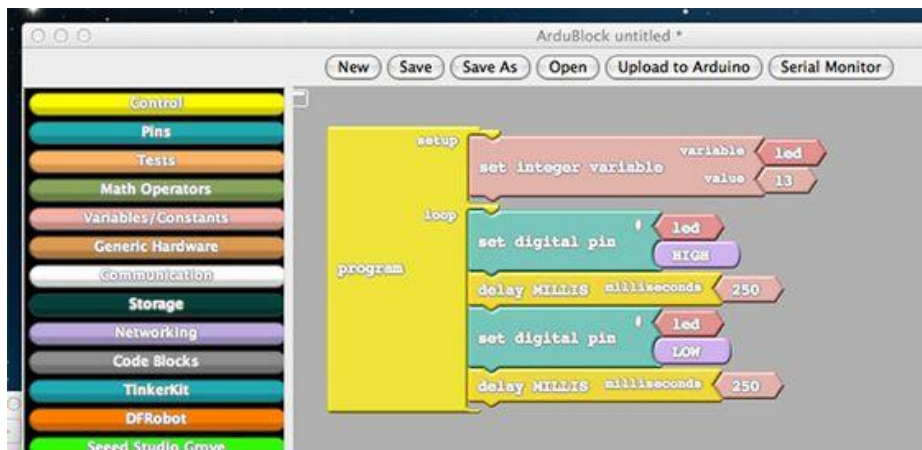


Fig. 3.4. ArduBlock.

- **Snap4Arduino:** se basa en el lenguaje de tomar y soltar (drag&drop) desarrollado en Berkeley. Este lenguaje ofrece una experiencia ligeramente diferente a la de ArduBlock. El método para armar sus scripts se parece mucho al diseñar visualmente una interfaz, pero claramente el lenguaje está diseñado para gente de más edad. Snap4Arduino (Fig. 3.5) no compila códigos, pero interactúa con el Arduino mientras está conectado a la computadora.

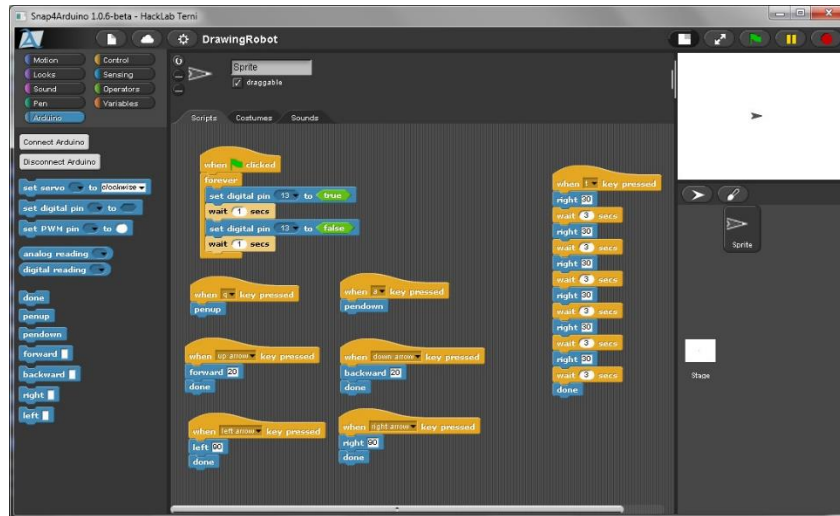


Fig. 3.5. Snap4Arduino.

- Python:** Aunque Python trabaja en general como un intérprete, es posible usarlo para comunicar el dispositivo a través del puerto serial. Esto se hace de forma muy simple en cualquier sistema Unix, pero si se usa PC o Mac, entonces Pyserial puede ser de mucha ayuda. Generalmente no se recomienda introducirse en el lenguaje Python si se es principiante.

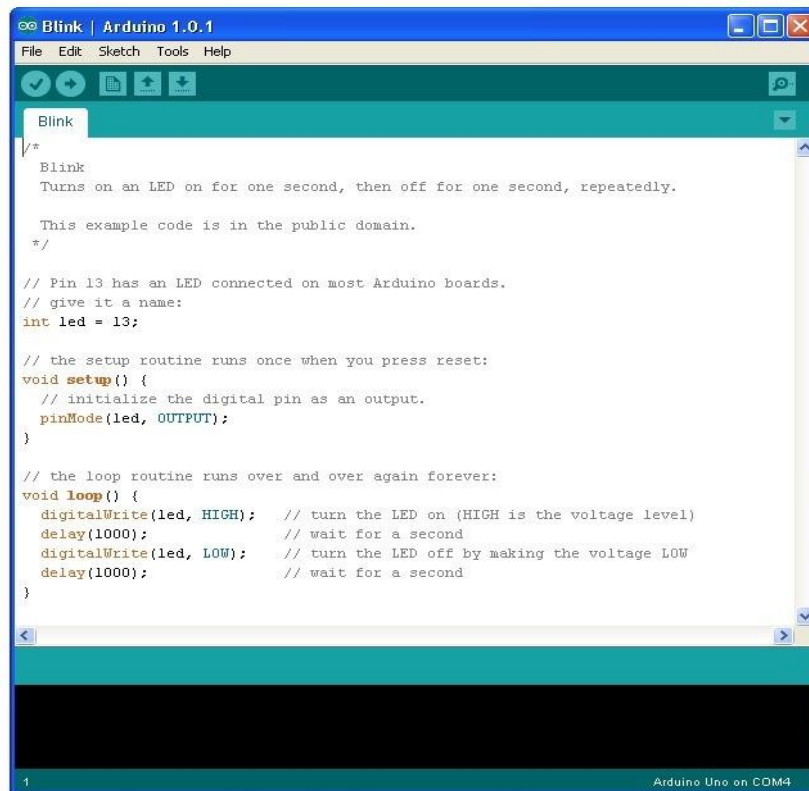
```

1  import json
2
3  data_store = 'todos.json'
4
5  class Todo:
6
7      def __init__(self, text):
8          self.text = text
9          self.isComplete = False
10
11      def complete(self):
12          self.isComplete = True
13
14
15  class Todos:
16
17      def __init__(self, data_store):
18          self.data_store = data_store
19          self.todos = []
20          with open(self.data_store) as data_file:
21              self.todos = json.load(data_file)['todos']
22
23      def add(self, text):
24          self.todos.append(Todo(text))
25          self.save()
26

```

Fig. 3.6. Python.

- **Sketch:** es uno de los lenguajes con más apoyo por parte de la comunidad de Arduino y se usa para un buen número de proyectos. Este lenguaje es muy parecido a C en su sintaxis y es muy sencillo de usar en general. Hay mucha documentación y es una posibilidad muy sólida frente a otros que aún tienen dificultades (Fig. 3.7).

The image shows a screenshot of the Arduino IDE interface. The window title is "Blink | Arduino 1.0.1". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". Below the menu bar is a toolbar with icons for saving, opening, and other functions. The main text area contains the following code:

```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 *
 * This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000);             // wait for a second
  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
  delay(1000);             // wait for a second
}
```

The status bar at the bottom indicates "1" on the left and "Arduino Uno on COM4" on the right.

Fig. 3.7. Sketch Arduino.



### 3.5.1.1 Estructura de programación IDE Arduino

La estructura básica del lenguaje de programación de Arduino es bastante simple y se compone de al menos dos partes. Estas dos partes necesarias, o funciones, encierran bloques que contienen declaraciones, estamentos o instrucciones (Fig. 3.8.).

```

void setup()
{
  estamentos;
}
void loop()
{
  estamentos;
}

```

Fig. 3.8. Estructura básica de programación.

En donde `void setup()` es la parte encargada de recoger la configuración y `void loop()` es la que contiene el programa que se ejecutara cíclicamente. Ambas funciones son necesarias para que el programa se ejecute correctamente.

La función de configuración debe contener la declaración de las variables. Es la primera función a ejecutar en el programa, se ejecuta solo una vez y se utiliza para configurar o inicializar `PinMode` (modo de trabajo de las I/O), configuración de la comunicación en serie y otras.

La función `loop` siguiente contiene el código que se ejecutara continuamente (lectura de entradas, activación de salidas, etc.) esta función es el núcleo de todos los programas de Arduino y la que realiza la mayor parte del trabajo.

- **Void Setup()**

La función `setup` (Fig. 3.9.) se invoca solo una vez cuando comienza el programa empieza. Se utiliza para inicializar los modos de trabajo de los pines, o puertos serie. Debe ser incluido en un programa, aunque no haya declaración que ejecutar.

```

void setup()
{
  pinMode(pin, OUTPUT); // configura el 'pin' como salida
}

```

Fig. 3.9. Ejemplo void setup().

- **Void loop()**

Después del setup(), la función void loop()(Fig. 3.10.) hace precisamente lo que sugiere su nombre, se ejecuta de forma cíclica, lo que posibilita que el programa este respondiendo continuamente antes los eventos que se produzcan en la tarjeta.

```

void loop()
{
  digitalWrite(pin, HIGH); // pone en uno (on, 5v) el 'pin'
  delay(1000); // espera un segundo (1000 ms)
  digitalWrite(pin, LOW); // pone en cero (off, 0v.) el 'pin'
  delay(1000);
}

```

Fig. 3.10. Ejemplo void loop().

- **PinMode(pin,Mode)**

Esta instrucción (Fig. 3.11.) es utilizada en la parte de configuración void setup() y sirve para la configurar el modo de trabajo de un PIN pudiendo ser INPUT(entrada) u OUTPUT(salida).

```

pinMode(pin, OUTPUT); // configura 'pin' como salida

```

Fig. 3.11. Ejemplo instrucción pinMode.

Los terminales de Arduino, por defecto, están configurados como entradas, por lo tanto, no es necesario definirlos en el caso de que vayan a trabajar como entradas. Los pines configurados como entrada quedan, bajo el punto de vista eléctrico como entradas en estado de alta impedancia.

Estos pines tienen a nivel interno una resistencia de  $20K\Omega$  a las que se puede acceder mediante software. A estas resistencias se accede de la manera como se ve en la Fig. 3.12.

```
pinMode(pin, INPUT); // configura el 'pin' como entrada
digitalWrite(pin, HIGH); // activa las resistencias internas
```

Fig. 3.12. Activación de resistencias internas

Las resistencias internas normalmente se utilizan para conectar las entradas a interruptores. En el ejemplo de la Fig. 3.12. no se trata de convertir un pin en salida, es simplemente un método para activar las resistencias interiores.

Los pines configurados como OUTPUT(salida) se dice que están en un estado de baja impedancia y pueden proporcionar 40mA de corriente a otros dispositivos y circuitos. Esta corriente es suficiente para alimentar un diodo LED, pero no es suficientemente grande como para alimentar cargas de mayor consumo como relés, solenoides o motores.

- **DigitalRead(pin)**

Lee el valor de un pin (definido como digital) dando un resultado HIGH(alto) o LOW(bajo). El pin se puede especificar ya sea como una variable o una constante (Fig. 3.13).

```
valor = digitalRead(Pin); // hace que 'valor' sea igual al estado leído
                          // en 'Pin'
```

Fig. 3.13. Ejemplo instrucción digitalRead.

- **DigitalWrite(pin, value)**

Envía al pin definido previamente como OUTPUT el valor HIGH o LOW (poniendo en 1 o 0 la salida). El pin se puede especificar ya sea como una variable o como una constante (Fig. 3.14.).

```
digitalWrite(pin, HIGH); // deposita en el 'pin' un valor HIGH (alto o 1)
```

Fig. 3.14. Ejemplo instrucción digitalWrite.

En la Fig. 3.15. Se presenta un ejemplo de las instrucciones mencionadas anteriormente, el cual consiste en leer el estado de un pulsador conectado a una entrada digital y lo escribe en el pin de salida led.

```
int led = 13; // asigna a LED el valor 13
int boton = 7; // asigna a botón el valor 7
int valor = 0; // define el valor y le asigna el valor 0

void setup()
{
  pinMode(led, OUTPUT); // configura el led (pin13) como salida
  pinMode(boton, INPUT); // configura botón (pin7) como entrada
}

void loop()
{
  valor = digitalRead(boton); //lee el estado de la entrada botón
  digitalWrite(led, valor); // envía a la salida 'led' el valor leído
}
```

Fig. 3.15. Ejemplo de programación.

- **AnalogRead(pin)**

Lee el valor de un determinado pin definido como entrada analógica como una resolución de 10 bits. Esta instrucción solo funciona en los pines analógicos de la placa. El rango de valor que podemos leer oscila de 0 a 1023 (Fig.3.16).

```
valor = analogRead(pin); // asigna a valor lo que lee en la entrada 'pin'
```

Fig. 3.16. Ejemplo instrucción analogRead.

- **AnalogWrite(pin, value)**

Esta instrucción sirve para escribir un pseudo-valor analógico utilizando el procedimiento de modulación de ancho de pulso (PWM) a uno de los pines de Arduino marcados como “pin PWM”. El valor que se puede enviar a estos pines de salida analógica puede darse en forma de variable o constante, pero siempre con un margen de 0-255 (Fig. 3.17.).

```
analogWrite(pin, valor); // escribe 'valor' en el 'pin' definido como
                           analógico
```

Fig. 3.17. Ejemplo instrucción analogWrite.

Si se envía el valor de 0 en la instrucción de la Fig. 20 se generará una salida de 0 volts en el pin especificado, por el contrario, si se envía un valor de 255 se genera una salida de 5 volts a la salida del pin especificado. Para valores de entre 0 y 255, el pin saca tensiones entre 0 y 5 volts. Teniendo en cuenta el concepto de señal PWM, por ejemplo, un valor de 64 equivaldrá a mantener 0 volts en tres cuartas partes del tiempo y 5 volts una cuarta parte del tiempo, un valor de 128 equivaldrá a mantener la salida en 0 volts la mitad del tiempo y 5 volts la otra mitad.

- **Delay(ms)**

Detiene la ejecución del programa la cantidad de tiempo en ms que se le indica en la propia instrucción. De tal manera que 1000 equivale a 1 segundo (Fig. 3.18).

```
delay(1000); // espera 1 segundo
```

Fig. 3.18. Ejemplo instrucción delay.

- **Monitor serial**

El monitor serial consiste en una consola de entrada y salida, esto quiere decir que podemos mostrar datos enviados por nuestra placa Arduino y también podemos enviar datos a nuestra placa. En la Fig. 3.19. Se muestra como acceder al monitor serial desde IDE de Arduino.

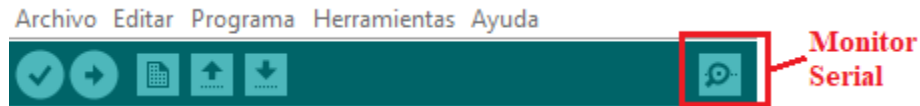


Fig. 3.19. Botón acceso monitor serial.

Para utilizar el puerto serial y fijar la velocidad en baudios para la transmisión de datos en serie. Se utiliza la instrucción de la Fig. 3.20. Las velocidades en baudios soportadas son de 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 o 115200.

```
void setup()
{
  Serial.begin(9600); // abre el Puerto serie
}                    // configurando la velocidad en 9600 bps
```

Fig. 3.20. Ejemplo instrucción activación monitor serial.

Para imprimir datos en el puerto serial se utiliza la instrucción de la Fig. 3.21. La instrucción actualiza automáticamente los datos recibidos y genera un salto de línea.

```
Serial.println(analogValue); // envía el valor 'analogValue' al puerto
```

Fig. 3.21. Ejemplo instrucción Serial.println.

En el siguiente ejemplo (Fig. 3.22.) se toma una lectura analógica desde el pin 0 y se envían los datos al monitor serial cada un segundo.

```
void setup()
{
  Serial.begin(9600); // configura el puerto serie a 9600bps
}

void loop()
{
  Serial.println(analogRead(0)); // envía valor analógico
  delay(1000); // espera 1 segundo
}
```

Fig. 3.22. Programa ejemplo monitor serial.

## Capítulo 4 . Comunicación Bluetooth

### 4.1 Comunicación Bluetooth y Arduino

Bluetooth es una especificación industrial para redes inalámbricas de área personal (WPAM) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM (Industrial, Scientific and Medical) de los 2.4 GHz. Los principales objetivos que se pretenden conseguir con esta norma son:

- Facilitar las comunicaciones entre equipos móviles.
- Eliminar los cables y conectores entre estos.
- Ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la sincronización de datos entre equipos personales.

Se denomina Bluetooth al protocolo de comunicaciones diseñado especialmente para dispositivos de bajo consumo, que requieren corto alcance de emisión y basados en transceptores de bajo costo.

Los dispositivos que incorporan este protocolo pueden comunicarse entre sí cuando se encuentran dentro de su alcance. Las comunicaciones se realizan por radiofrecuencia de forma que los dispositivos no tienen que estar alineados y pueden incluso estar en habitaciones separadas si la potencia de transmisión es suficiente. Estos dispositivos se clasifican como “clase 1”, “clase 2” o “clase 3” en referencia a su potencia de transmisión.

Los dispositivos con Bluetooth también pueden clasificarse según su capacidad de canal, lo que se demuestra en la Fig. 4.1.



Versión	Ancho de banda
Versión 1.2	1 <a href="#">Mbit/s</a>
Versión 2.0 + EDR	3 <a href="#">Mbit/s</a>
Versión 3.0 + HS	24 <a href="#">Mbit/s</a>
Versión 4.0	32 <a href="#">Mbit/s</a>

Fig. 4.1. Clasificación Bluetooth.

La especificación de Bluetooth define en canal de comunicación a un máximo de 720 Kbit/s (1 Mbit/s de capacidad bruta) con un rango de 10 metros. Opera en una frecuencia de radio de 2,4 a 2,48 GHz con amplio espectro y saltos de frecuencia con posibilidad de transmitir en Full Duplex con un máximo de 1600 saltos por segundo. Los saltos de frecuencia se dan entre un total de 79 frecuencia con intervalos de 1 MHz esto permite dar seguridad y robustez.

El hardware que compone el dispositivo Bluetooth está compuesto por dos partes:

- Un dispositivo de radio, encargado de modular y transmitir la señal.
- Un controlador digital, compuesto por una CPU, un procesador de señales digitales (DSP-Digital Signal Processor) llamado Link Controller o contador de Enlace y de las interfaces con el dispositivo anfitrión.

El LC o Link Controller se encarga del procesamiento de la banda base y del manejo de los protocolos ARQ (Automatic Repeat-reQuest) y FEC (Forward Error Correction) de la capa física, además, se encarga de las funciones de transferencia tanto asincrónica como sincrónica, la codificación de audio y cifrado de datos.

Los dispositivos Bluetooth pueden actuar como Masters o como Slaves. La diferencia es que un Bluetooth Slave solo puede conectarse a un Master y a nadie más, en cambio un Master Bluetooth, puede conectarse a varios Slaves o permitir que ellos se conecten para recibir y solicitar información de todos ellos, vigilando las transferencias de información (hasta un máximo de 7 Slaves). Esto se ejemplifica en la Fig. 4.2.

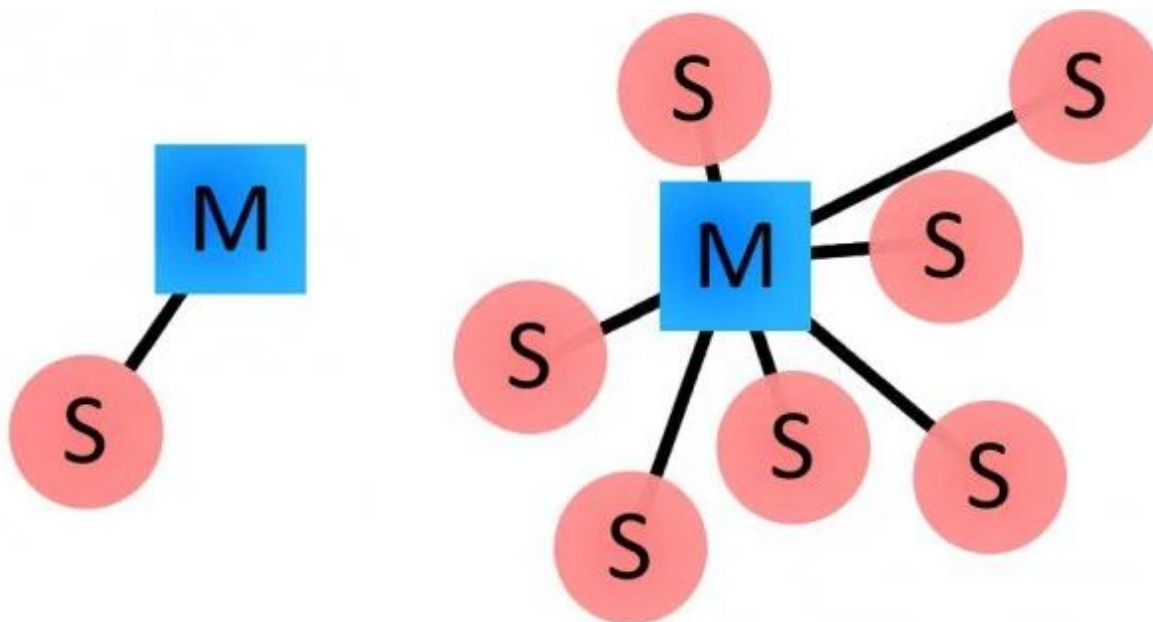


Fig. 4.2. Bluetooth slave vs Bluetooth Master.

Cada uno de los dispositivos que se identifican vía Bluetooth presenta una dirección única de 48 bits y además un nombre de dispositivo que nos sirva para identificarlo cómodamente a los usuarios.

Para dotar de comunicación Bluetooth a Arduino podemos hacerlo de varias formas:

- Módulo Bluetooth externo
- Módulo HC-05 o HC-06
- Módulo Bluetooth 4.0 HC-08 y HC-09
- Módulo Sparkfun
- Módulo integrado en placa como el Arduino BT
- Microcontrolador con Bluetooth integrado como el Arduino 101
- Shield Bluetooth
- Arduino Wireless programmer

Los módulos utilizados con más frecuencia y de mayor acceso en el mercado son los módulos HC-06 y HC-05 que son muy económicos y están disponibles independientes o en modo SHIELD.

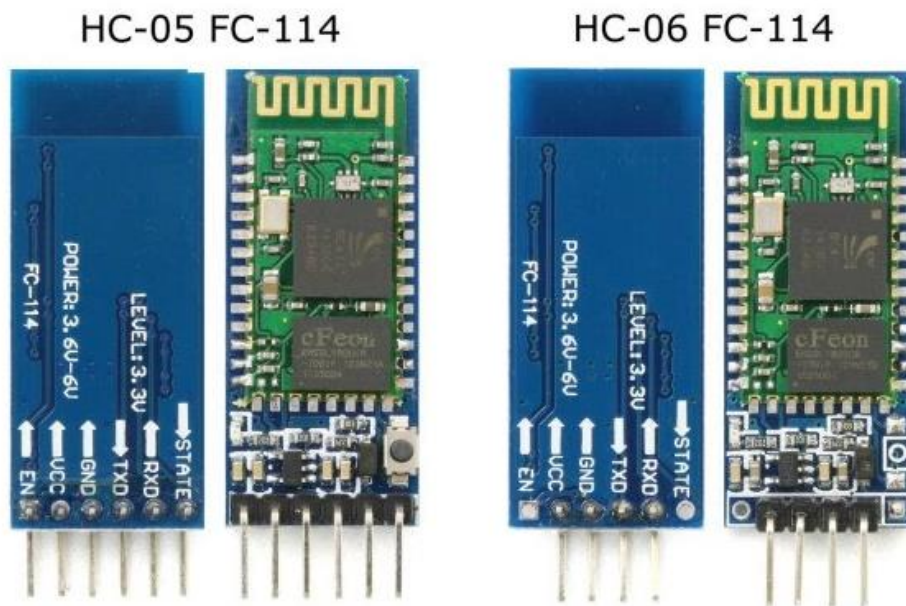


Fig. 4.3. Módulos comunicación Bluetooth.

Para utilizar el módulo HC-05 se requiere el uso de un puerto serie de la placa Arduino. Por lo tanto, mientras se una el modulo no se puede usar puerto serie en las placas modelo Uno, Mini y Nano. En el Arduino Mega no se tiene este problema, ya que incorpora 4 puertos serie.

Es necesario saber que cuando se esté cargando un nuevo programa a la placa Arduino se debe desconectar el modulo Bluetooth, dado que la programación se realiza a través del puerto serie, esta condición se da para todas placas Arduino.

Si realmente necesitamos ambas comunicaciones podemos emplear la **librería SoftSerial** para establecer una comunicación de puerto serie por cualquier pareja de pines digitales, aunque ello supondrá un coste adicional de tiempo de procesamiento en Arduino.

La conexión es sencilla. El módulo HC-05(Fig. 4.4.) se alimenta mediante Vcc y GND. Posteriormente conectamos el TXD y RXD a los opuestos de la placa Arduino (Fig. 4.5.).



Fig. 4.4. Pines HC-05.

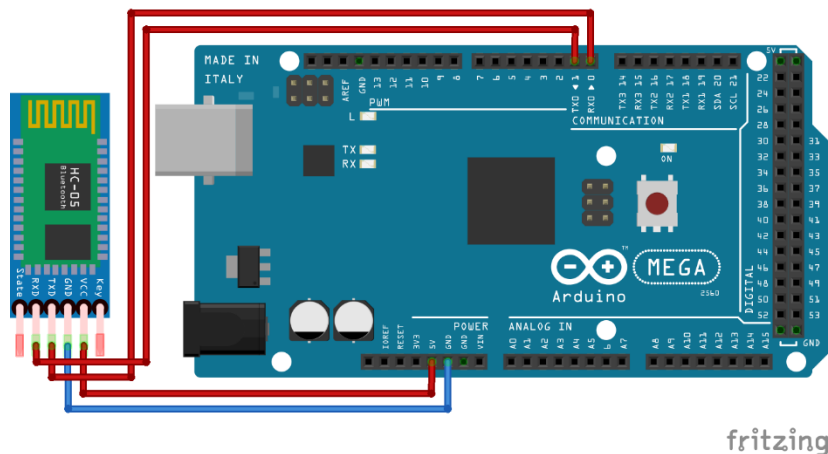


Fig. 4.5. Conexión HC-05 Arduino.

## 4.2 Conexión Bluetooth sistema Android

Para lograr una interacción con el modulo Bluetooth HC-05(Fig. 4.4) es necesario desarrollar una aplicación soportada en sistema Android. Para ello existen plataformas gratuitas de desarrollo como lo es App Inventor.

App inventor es un entorno de desarrollo de software creado por Google Labs para la elaboración de aplicaciones destinadas al sistema operativo Android. El usuario puede de forma visual y a partir de un conjunto de instrucciones básicas, ir ensamblando una serie de bloques para crear la aplicación.

Los proyectos generados a través de esta herramienta se almacenan automáticamente en los servidores de App inventor, permitiendo llevar en todo momento un seguimiento y control de todo el trabajo realizado.

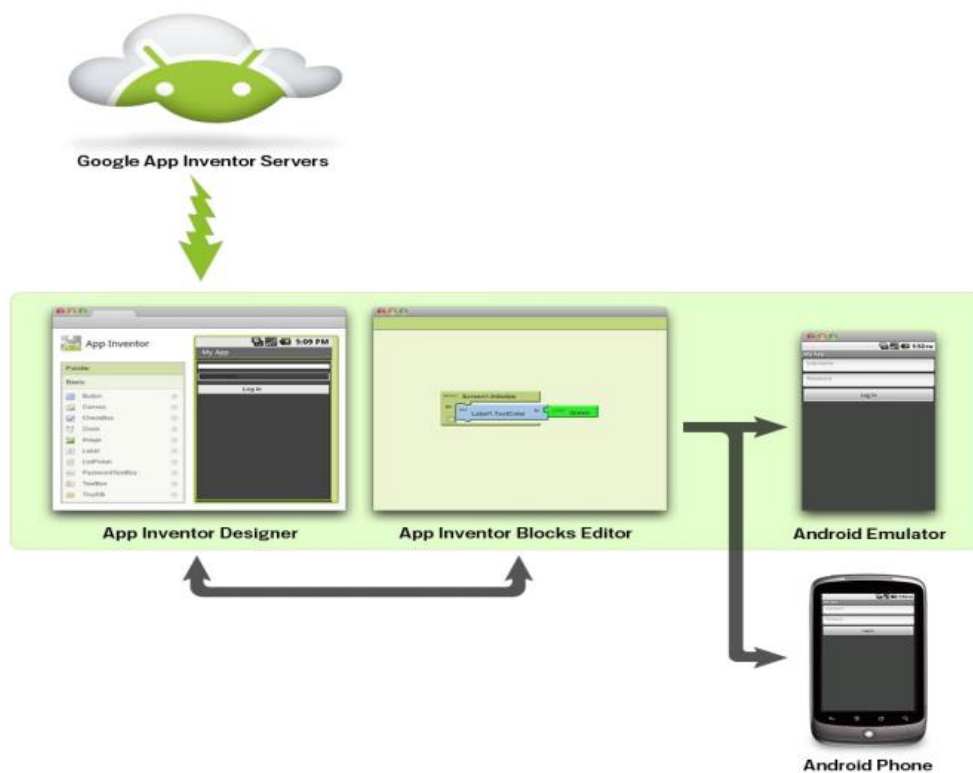


Fig. 4.6. Visión global App Inventor.

### 4.3 Arquitectura aplicación App Inventor

Las aplicaciones construidas mediante App Inventor están compuestas por los elementos que se muestran en el diagrama de la Fig. 4.7.

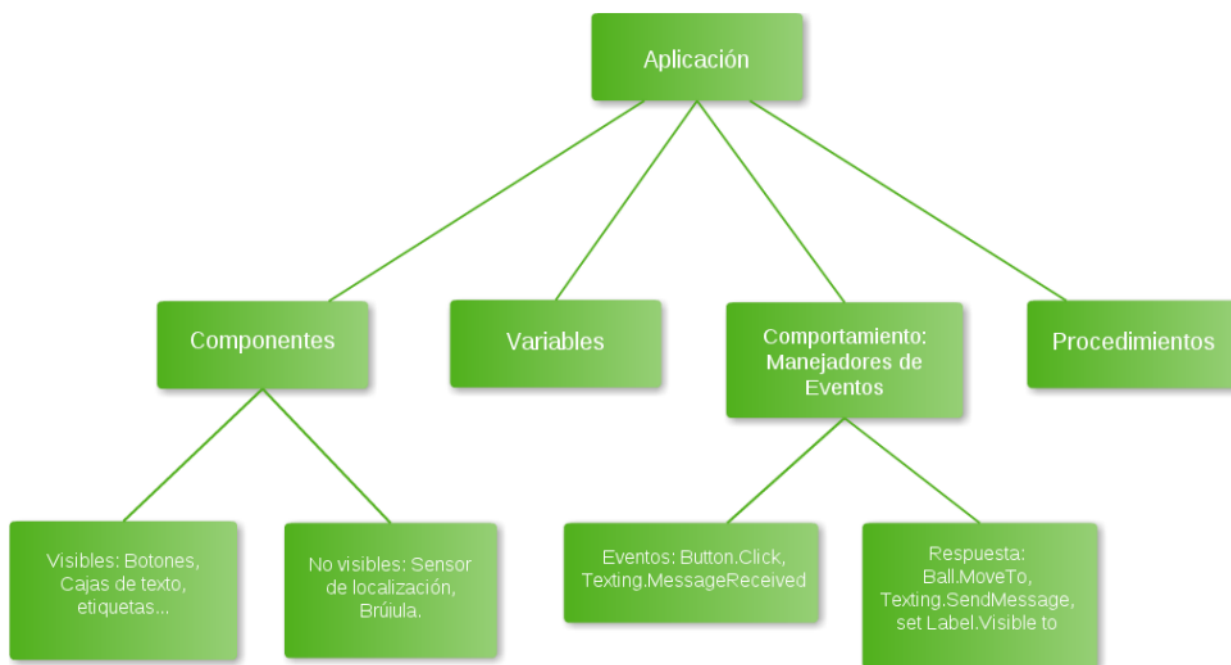


Fig. 4.7. Arquitectura interna aplicación creada por App Inventor.

Una buena manera de entender una aplicación, es descomponerla en dos partes, por un lado, los componentes y por otro lado los comportamientos.

### **4.3.1 Componentes**

Hay dos tipos de componentes principales en cualquier aplicación: los visibles y los no visibles.

Los componentes visibles son aquellos que podemos ver una vez hemos ejecutado nuestra aplicación (botones, cajas de texto, etiquetas, etc.). El conjunto de estos elementos se denomina comúnmente como la interfaz de usuario de la aplicación.

Por otro lado, los componentes no visibles son aquellos que no podemos ver en la aplicación, ya que no son parte de la interfaz de usuario. Proporcionan acceso a la funcionalidad interna de los dispositivos; por ejemplo, el componente “Texting” permite enviar y procesar mensajes de texto, y el componente “LocationSensor” permite determinar la localización del dispositivo. Ambos componentes están definidos mediante una serie de propiedades. Las propiedades son fragmentos de memoria que permiten almacenar información relativa al componente al que referencian. Los componentes visibles, por ejemplo, disponen de propiedades relativas a su posición, altura y anchura, y alineación, que definen conjuntamente su aspecto dentro de la aplicación global. Todas estas propiedades se definen dentro del diseñador de componentes de App Inventor.

### **4.3.2 Comportamiento**

El comportamiento define como una aplicación debe responder ante una serie de eventos, los producidos por la interacción del usuario (un clic de botón) y los externos (un SMS recibido en nuestro dispositivo). En este punto es donde reside la mayor complejidad en el desarrollo de aplicaciones. Afortunadamente, App Inventor proporciona un lenguaje visual de bloques que nos permite definir comportamientos de una forma muy precisa. Normalmente, podemos identificar el desarrollo de aplicaciones con la elaboración de una “receta”, es decir, siguiendo una secuencia lineal de instrucciones.

Sin embargo, la mayoría de las aplicaciones actuales, no cumple estrictamente este tipo de paradigma. No se ejecutan una serie de instrucciones en un orden predeterminado, sino que, la aplicación reacciona a una serie de eventos, normalmente iniciados por el usuario final de la aplicación. Por ejemplo, si el usuario hace clic sobre un botón, la aplicación responde realizando alguna operación (enviar un mensaje de texto, confirmar una determinada operación, etc.). Este tipo de aplicaciones se pueden interpretar como un conjunto de componentes que reaccionan ante unos determinados eventos. Las aplicaciones incluyen una serie de “recetas” (secuencias de instrucciones), las cuales se ejecutan cuando se producen los eventos asociados a las mismas.

### 4.3.3 Eventos

Los eventos inicializados por el usuario son el tipo más común. Reflejan la interacción del usuario final con la aplicación. Por ejemplo, en la Fig. 4.8 podemos observar cómo se ha definido que cuando el usuario hace click sobre el “Button25” se envía un texto a través de la conexión bluetooth con la letra “A”.

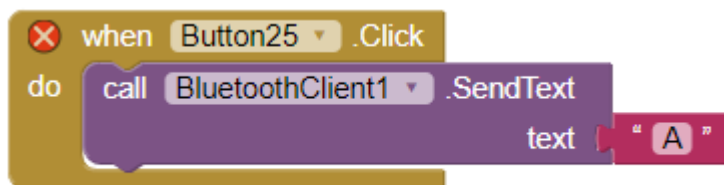


Fig. 4.8. Instrucción para enviar textos.



## **4.4 Entorno de desarrollo**

El entorno de programación de App Inventor tiene tres partes fundamentales:

- El Diseñador de Componentes se ejecuta en el navegador web. Es utilizado para seleccionar los componentes de nuestra aplicación y especificar sus propiedades. A través de él, iremos definiendo el aspecto visual.
- El Editor de bloques se ejecuta en una ventana independiente del Diseñador de Componentes. Nos permitirá crear los comportamientos necesarios de nuestra aplicación y asociarlos a sus respectivos componentes.
- Un dispositivo Android nos permitirá ejecutar y comprobar nuestras aplicaciones mientras las estamos desarrollando. Si no disponemos de ningún dispositivo adecuado, podremos probar nuestras aplicaciones utilizando el emulador de Android, el cual viene integrado dentro del sistema.

### **4.4.1 Diseñador de componentes**

Los componentes son los elementos que combinamos para crear nuestras aplicaciones. Algunos son muy simples, como por ejemplo una etiqueta que muestra un texto en la pantalla, o un botón, mediante el cual podemos iniciar una determinada acción. Otros componentes son más elaborados; un componente “Canvas” que permite visualizar imágenes o animaciones; el acelerómetro, un sensor de movimiento que detecta cuando movemos o agitamos el teléfono; o componentes que nos permiten enviar mensajes de texto, reproducir música y video, obtener información desde páginas web, etc.

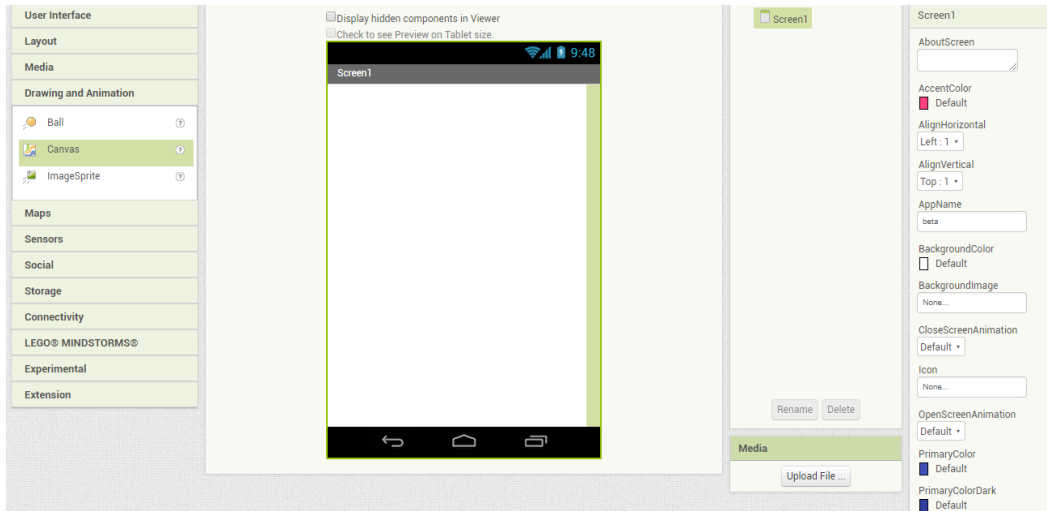


Fig. 4.9. Ventana principal diseñador de componentes.

#### 4.4.2 Editor de bloques

El Editor de Bloques (Blocks Editor) nos permite añadir y asignar tareas específicas a los componentes individuales de nuestra aplicación, de tal modo que podamos ir creando de manera conjunta la funcionalidad global que pretendemos que tenga nuestra aplicación.

El Editor de Bloques está implementado como una aplicación de Java Web Start, que se ejecuta en nuestro ordenador.

Para iniciarlo, debemos hacer clic sobre el botón “Blocks” situado en la parte superior derecha del Diseñador de Componentes. Una vez pulsado, el texto del botón cambiará y nos indicará que se está cargando el Editor de Bloques (Fig. 4.10.).

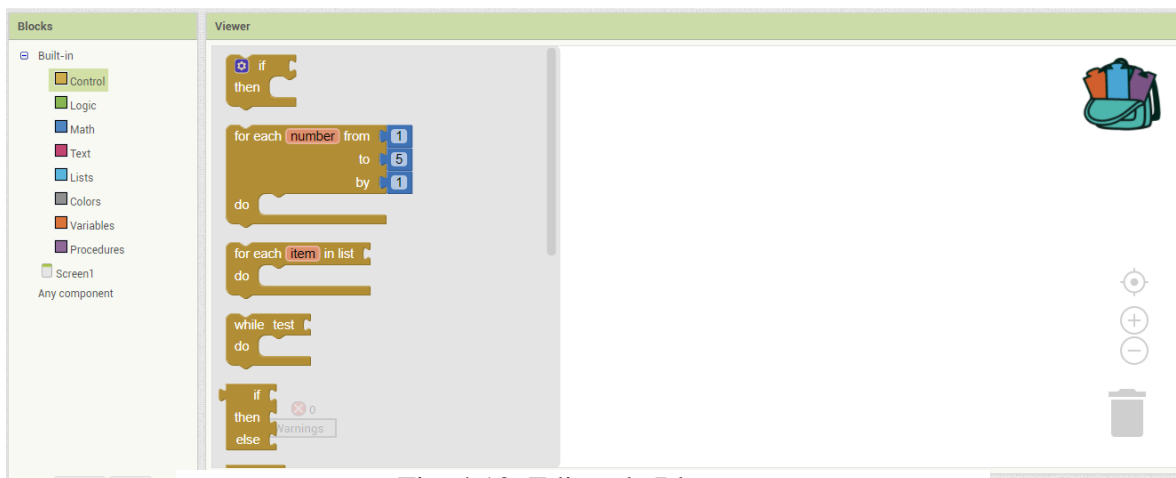


Fig. 4.10. Editor de Bloques.

## Capítulo 5 . Control Domótico

### 5.1. Domótica

Teniendo claro el concepto de Domótica, que involucra a un hogar inteligente que pueden funcionar de forma automática o ser autosuficiente, sin la necesidad de una supervisión de carácter humano, ya sea directa o por medio de una regulación indirecta. Gracias a los nuevos tiempos, la domótica ha adquirido popularidad en el mundo, ya que todas las personas prefieren la comodidad, la seguridad, confort y máxima interacción o accesibilidad, en la cual, sea la tecnología lo que predomine. El motivo de éxito de esta tecnología fue gracias a los estereotipos del futuro, planteados aproximadamente en el año 1970 a 1980, en donde se ejemplificaban sistemas robotizados que proporcionaban autonomía en el área de la limpieza, cocina y quehaceres del hogar.

La domótica en si tiene numerosos exponentes mundiales que entregan sus servicios a cambio de la modernización del hogar, entre las más destacadas se encuentran IECOR, ubicada en Córdoba, INGENIUM, con sedes a lo largo del mundo, y otras empresas nacionales, tales como BTICINO o TECHOME. La ventaja de este negocio en rigores más profundos y a diferencia de tener una vivienda normal son las mejoras o actualizaciones de acciones propias que son recurrentes en el día a día. Entre las mejoras destacadas podemos encontrar:

- **Comodidad:** Está la posibilidad de gestionar toda la actualización de vivienda en una aplicación para teléfonos móviles o tabletas con opciones de control remoto, por lo que genera una opción de libertad al momento de accionar luces o elementos electrónicos.
- **Accesibilidad:** Mediante el constante monitoreo y la implementación de la app móvil, se puede accionar cualquier variable desde cualquier punto del hogar.
- **Seguridad:** Con la implementación de luces de emergencia, luces de alarma y bocinas anti intrusión, la modernización de la casa prioriza seguridad ante cualquier situación.
- **Ahorro energético:** En base a los datos proporcionados por los sensores involucrados, para el sistema es sencillo accionar calefactores o luces automáticamente cuando sea necesario en caso de ausencia humana.

## 5.2. Estructuración de un control domótico

La estructura base para el control domótico son los elementos esenciales que logran gestionar el comportamiento de nuestro hogar, que a su vez pertenecen al sistema global. Como en cualquier tipo de control se denotan elementos típicos como lo son los controladores, interfaces de control, actuadores y sensores.

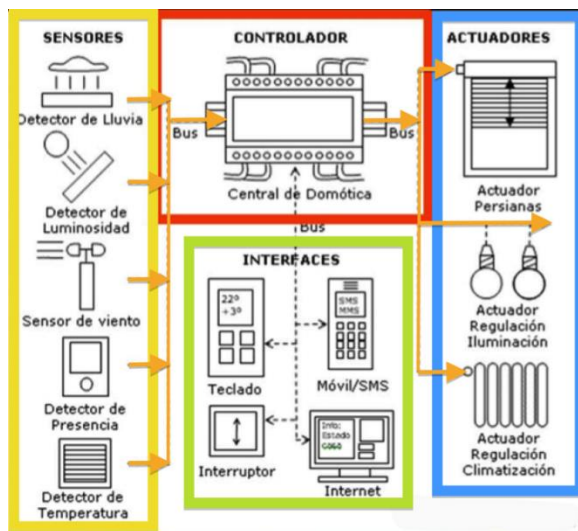


Fig. 5.1. Estructura Sistema Domótico

Siendo el cuadro rojo (controlador) como la base del sistema, el cual se encarga de interpretar señales provenientes de los sensores, las cuales pueden ser de carácter binario o variables físicas como humedad, temperatura, que son reguladas para estabilizar las condiciones designadas y aptas para el hogar.

En el cuadro amarillo se encuentran los sensores básicos para la implementación de la tecnología en un hogar, siendo estos los que transmiten la información al controlador, mediante cableado o conexión inalámbrica.

Mediante las interfaces del cuadro verde se puede gestionar la acción requerida que se ve reflejada en los actuadores del cuadro azul, quienes ejecutan la acción procesada por el controlador.

### 5.3. Transferencia de datos

Según el traspaso de información, varían la forma es que se transmiten los datos, precisamente por esto es que nombran a continuación dos formas en las cuales se logra la comunicación:

- **Mediante cables:** utilizando conexiones basadas en cableados entre componentes se puede lograr la transferencia de datos referentes al medio ambiente. Los cables tradicionales como el de red eléctrica o el cable telefónico son aptos para la transmisión de datos.
- **Inalámbricamente:** Para la inclusión de un sistema domótico refinado, vale destacar que una comunicación inalámbrica es un escenario perfectamente accesible en la actualidad. Comunicación bluetooth, wifi, infrarrojo o Xbee son compatibles y aptas en la comunicación.

### 5.4. Arquitectura del sistema domótico

Debido a la variedad de formas en la que un sistema de control puede ser conectado para la comunicación, es que se desarrollan las arquitecturas de control, que consisten en especificar un diagrama de control basado en las necesidades del medio que requiere ser controlado, indicando variables predominantes

Como condición, el sistema a controlar debe tener un mínimo de estabilidad robustez necesaria, con el objetivo de que pueda desenvolverse frente a perturbaciones o problemas del ambiente. Su funcionalidad debe ser de carácter lineal y sin variaciones bruscas en sus actuadores, ya que puede significar daño al equipo o a las propias personas que lo estén manipulando.

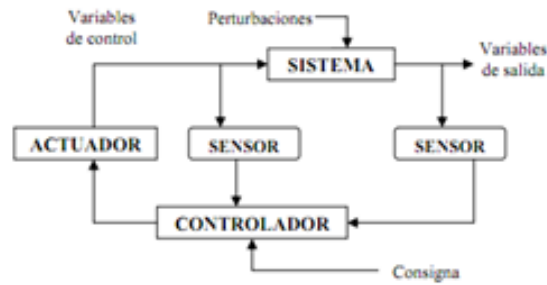


Fig. 5.2. Arquitectura genérica de un sistema domótico

Dentro del área de la domótica, se pueden encontrar 4 tipos de arquitecturas que permiten al sistema manejar las variables requeridas en una vivienda. Los siguientes tipos son:

#### 5.4.1. Arquitectura con foco central

La característica principal de la arquitectura centralizada es la presencia de un controlador central que organiza todo el funcionamiento del sistema, además establece las acciones a tomar a partir de la información enviada por los sensores del sistema.

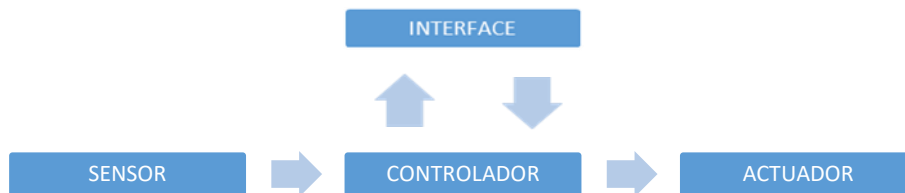


Fig. 5.3. Arquitectura con foco central

La arquitectura centralizada permite la opción de tener un solo dispositivo con controlador e interfaz de mando incluidas, lo que facilita la instalación. Al ser un solo dispositivo, la planificación del sistema se vuelve innecesaria ya que la información se envía al mismo dispositivo, el cual recibe, interpreta y gestiona. A pesar de esto y técnicamente hablando, la falencia de esta arquitectura radica en que al ser responsable un solo controlador, se le atribuyen todas las fallas en caso que hubiera, por lo tanto todo el sistema domótico se vería involucrado. Para evitar esto, se debe tener un controlador con mayor capacidad o más entradas para todas las variables.

### 5.4.2. Arquitectura Distribuida

En esta arquitectura está la opción de que cada actuador, sensor o interface pueda tener su propio controlador interno, o externos pero conectados directamente a un bus. Debido a esto, la comunicación en este tipo de arquitectura se realiza de forma directa sin recurrir al uso de un solo controlador.

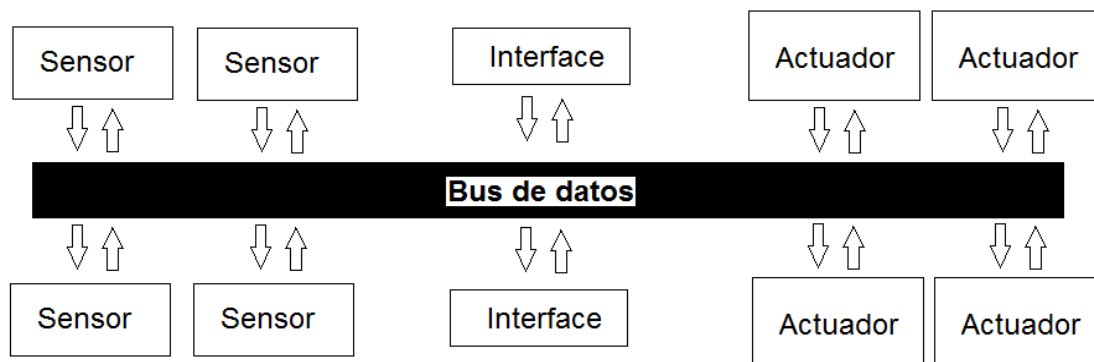


Fig. 5.4. Modelo de arquitectura distribuida

Esta arquitectura tiene similitudes con la centralizada, variando en que cada elemento propio puede manejar acciones simples de forma automática o mediante control. Por esta razón cada unidad actúa como sensor o actuador según se necesite, por lo que no hacen falta elementos de precios tan excesivos, en comparación a otras arquitecturas.

La ventaja de esta arquitectura es que, si ocurre un fallo de algún elemento del sistema, como una puerta o una luz, no afecte al resto de los procesos que están involucrados en la misma zona o habitación. Mientras que la desventaja visible es la dificultad de implementación de esta arquitectura, ya que se necesita un análisis profundo del hogar para implementar la tecnología de control, además de identificar el método óptimo de comunicación para la interacción de los dispositivos comprometidos.

### 5.4.3. Arquitectura sin foco central

Caracterizada por no tener un controlador central debido a que posee varios controladores para la regulación de las determinadas variables existentes.

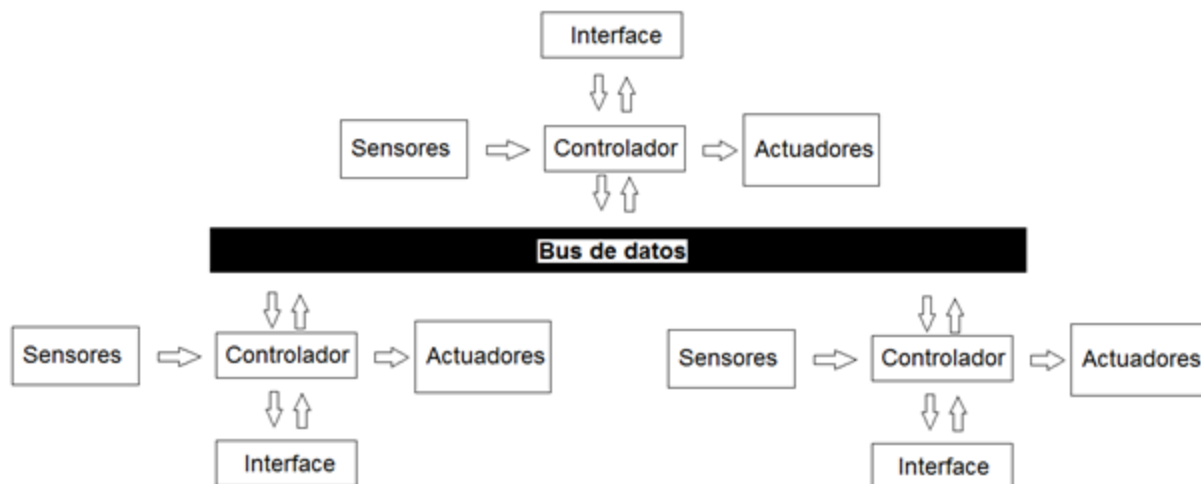


Fig. 5.5. Modelo Arquitectura Descentralizada

Su función es dividir las distintas zonas de ejecución del sistema, ya sea iluminación, apertura de puertas o la misma seguridad. La idea de esta arquitectura es lograr una distribución en el sistema teniendo como principal soporte la variedad de controladores, capaces de enviar información reduciendo las probabilidades de fallas de carácter doméstico, de esta forma se ve reflejada una mejora en la seguridad en comparación a otras arquitecturas.

La desventaja es que cada controlador debe manejar un número reducido de sensores y actuadores, para evitar fallas pertinentes además de poseer algoritmos de control para proteger a los elementos involucrados.

Este sistema permite ahorrar costos debido a que al ser particionado no se necesitan de controladores de gran nivel de procesamiento.



#### 5.4.4. Arquitectura Mixta

Consiste en una mezcla de arquitecturas usándolas según convenga, mediante adaptación de situación de control. Puede poseer un controlador central, como también múltiples controladores descentralizados junto a elementos como sensores y actuadores.

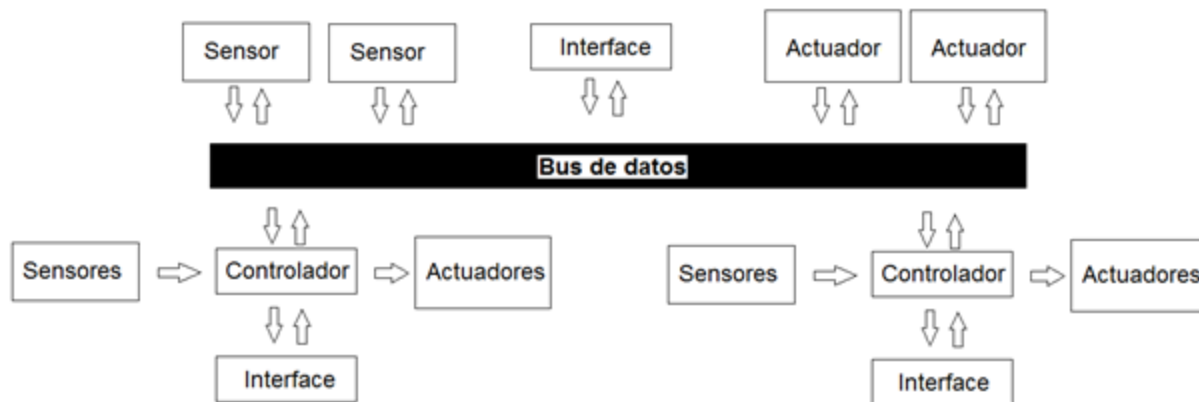


Fig. 5.6. Modelo Arquitectura Mixta

Una arquitectura intuitiva, en la que se implementa el modo que mejor se ajuste al sistema en cuestión, donde se permite la utilización de una por separada o en conjunto dependiendo de costos, rapidez y complejidad con la que se desee hacer el proyecto. Debido a esto no hay una brecha clara que separe esta arquitectura de una distribuida o descentralizada.

Dada la conformación inusual de esta arquitectura, se puede divisar una estructura jerárquica clara y definida, siendo el controlador el punto de tope y debajo de este se encontrarían los sensores actuadores y controladores de menor nivel.

Como es una arquitectura combinada, posee las mismas ventajas y desventajas de las arquitecturas específicas.

## Capítulo 6 . Implementación

Lo esencial al momento de implementar un sistema domótico, es que se busque la automatización y autonomía completa al menor costo posible. Gracias a la plataforma Arduino, las variaciones de precios en la instalación de estos sistemas son entre los \$65500 hasta lo más exuberante que rodea los \$300000 considerando actuadores y sensores. A pesar de una relativa facilidad al implementar esto, hay que poner atención en las frecuencias proporcionadas por Arduino y las características físicas del chip ATmega, ya que limitan las posibilidades.

### 6.1. Primera Fase

Ante cualquier proyecto de ensamblaje de un circuito, se debe probar y testear en una placa base de prueba o práctica, y este sistema no es la excepción. De esta forma se corrobora el buen funcionamiento del diseño principal y la elección de los elementos pertenecientes sin riesgo de fallas o daños físicos a estos mismos.

El circuito inicial que representa la iluminaria del sistema hogar está representado por 13 leds de color blanco de alta luminiscencia, distribuidos entre habitaciones y conectados cada uno a una resistencia respectiva por seguridad. Aparte se tienen leds que de alarma compuestos por una tira de leds azules de 12 (V); 20 (mA) que se activan mediante un sensor de movimiento, que a su vez también gatilla la alarma incorporada en el sistema, generada por un buzzer.

Todo el prototipo está pensado para ámbitos educativos y aplicados a la maqueta física construida. En el caso de querer llevar esta idea a un hogar, se debe reemplazar los sensores por barras laser o múltiples sensores de movimiento, de esta forma, cualquier irrupción sería advertida por el buzzer y la intermitencia de los leds de alarma. Además de seguridad, se presenta en el prototipo un sensor de temperatura que sensa y envía la información al controlador, lo que a su vez se muestra en un display conectado al sistema. Se señala la temperatura en grados Celsius.

También se ha agregado “aire acondicionado” simulado por un ventilador simple de 12 (v), mientras que la apertura de puertas se realiza mediante servomotores de voltaje medio 5 (v).

## **6.2. Arduino y Actuadores y Sensores**

Teniendo un conjunto de actuadores formados principalmente por 2 relés normalmente cerrados, lo que permite la comunicación a los sensores y al Arduino mismo. Uno de estos relés está comunicado directamente con la línea de leds de alarma de 12 (v), mientras que el otro va dirigido al ventilador, además un pin de cada relé se conecta en una fuente de alimentación de 12 Vdc que otorga el funcionamiento de estos dispositivos. Los relés van conectados en el Arduino en los pines 31 y 43.

Los servomotores y el buzzer se conectan directamente al Arduino, siendo los pines 49 y 51 de los servomotores y 9 el buzzer. El funcionamiento de la vigilancia es automático, no se necesita estar activando la alarma para que suene, sino que se activa la autonomía. Una vez activada la autonomía, cualquier movimiento que se interponga en el espectro de visión de la vigilancia, activará el buzzer y las luces de alarma, otorgando aviso al propietario del hogar. Para los servomotores basta con un botón de activación en la plataforma móvil para que la puerta se abra o se cierre.

Las variables medibles que se pueden encontrar en el ambiente, al menos la más importante es la temperatura. La lectura de esta variable nos permitirá interactuar con ella para obtener la cantidad óptima de condición, accionando el aire acondicionado o en caso de bajas temperaturas, con aperturas de ventanas o puertas. Para medir esto se necesitó un sensor de temperatura.

### 6.3. Plano de Maqueta

Con un prototipo de circuito realizado y funcionando se comienza con la elaboración del modelo de hogar en el cual actuará nuestro sistema. Se decidió realizar un modelo a escala de un hogar real, sus planos como referencia. Utilizando madera reciclada, se construyen paredes con agujeros cuadrados y rectangulares, que representan las ventanas y puertas. Construido en una base de la misma madera que genera soporte a la estructura.

Para el diseño de la maqueta se realizó el modelo en el programa computacional AutoCAD en el cual se especifican las medidas de la maqueta y de cada habitación. El hogar consta de un living-comedor, una cocina, un baño, dos dormitorios, un garaje y una habitación desván en la cual estarán todos los cables y controladores.

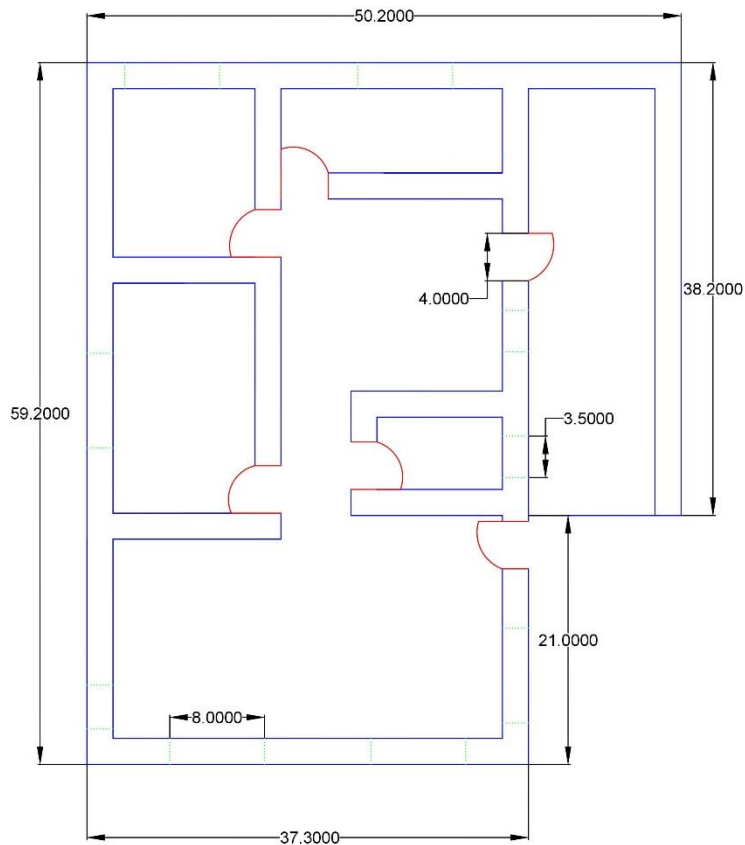


Fig. 6.1. Plano Hogar en AutoCAD

## 6.4. Plano Eléctrico

Siguiendo la línea de diseños realizados en el proyecto, se continuaría por el modelo del conexionado eléctrico realizado en la maqueta, en donde se demuestran los componentes que actúan en dicho conexionado. Inicia con la alimentación, que aporta suministro a la bifurcación entre los relés, la tira de leds, el ventilador y el modulo bluetooth conectado al Arduino MEGA. En la fig. 6.2 se puede apreciar el modelo. Los cables van directo en el Arduino y por sobre la maqueta para impedir la saturación de las habitaciones y exceso de cables a la vista.

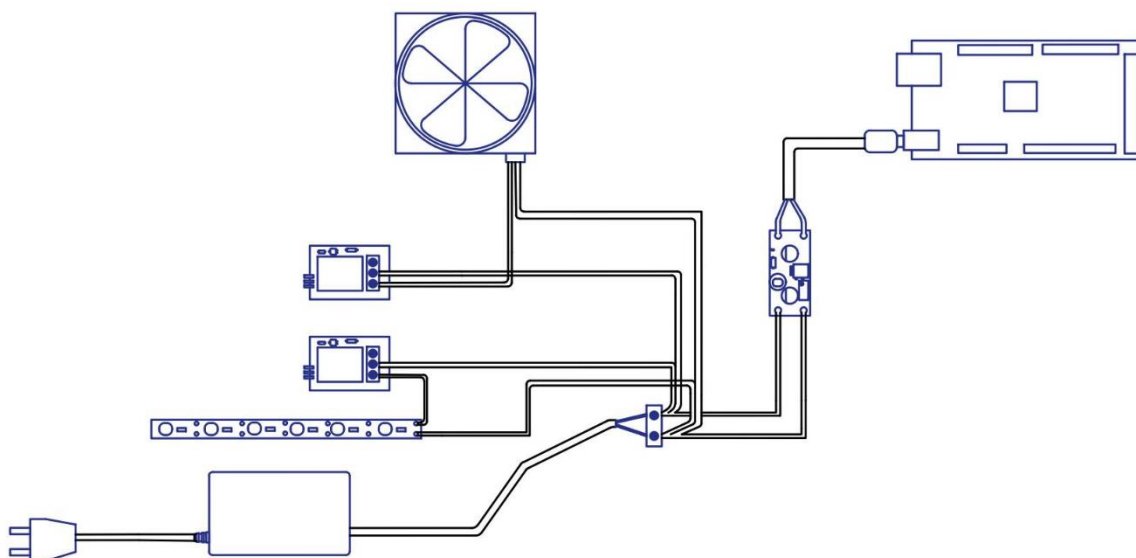


Fig. 6.2. Conexionado Eléctrico de la maqueta

Cabe mencionar que conectado al Arduino va también el display correspondiente que interactúa con el sensor de temperatura. La mayoría de componentes de este diseño van en la habitación especial, excepto el ventilador que se ubica en el living-comedor y la tira de leds, que abarca cocina y living.

## 6.5. Interfaz Aplicación Móvil

El interfaz es el medio por el cual el usuario se comunica con el sistema domótico. Gracias a esto, el usuario puede manipular la información y adaptarla a sus propias necesidades. La aplicación como se mencionó anteriormente, actúa como control para las diversas opciones de personalización, mediante el módulo bluetooth.

Al ser un proyecto con carácter educativo, se debe hacer una aplicación intuitiva y de fácil manejo, que manipule las variables directamente, ya que se enfoca en jóvenes que necesiten instrucción del tema. Tomando en cuenta esto, se priorizaron opciones básicas de on/off, es decir, interruptores que señalan el encendido y el apagado. No se introdujeron imágenes más que la de fondo, lo que genera una rapidez en la aplicación.

Como pantalla inicial se presenta una imagen referencial de la facultad de ingeniería y un acceso que da con la opción de control. Una vez ingresados en la opción de control se observan diversas opciones representadas por un color situacional, en donde el rojo representa los elementos apagados mientras que el verde los que están actualmente encendidos



Fig. 6.3. Portada Aplicación Móvil



Fig. 6.4. Pantalla de opciones

## Capítulo 7 . Ejecución del Sistema

Con todo este proceso de diseñar, programar, construir, probar y corregir, corresponde notificar el avance obtenido en el trayecto del proyecto, y para eso se necesita corroborar separadamente cada elemento del sistema.

### 7.1. Conexión Físico

Diseñado y completado el modelo de nuestro proyecto, se reflejaría en la Fig. 46, donde se aprecian las diversas conexiones que posee la maqueta. También se observan los elementos enlazados al controlador, como los sensores y actuadores.

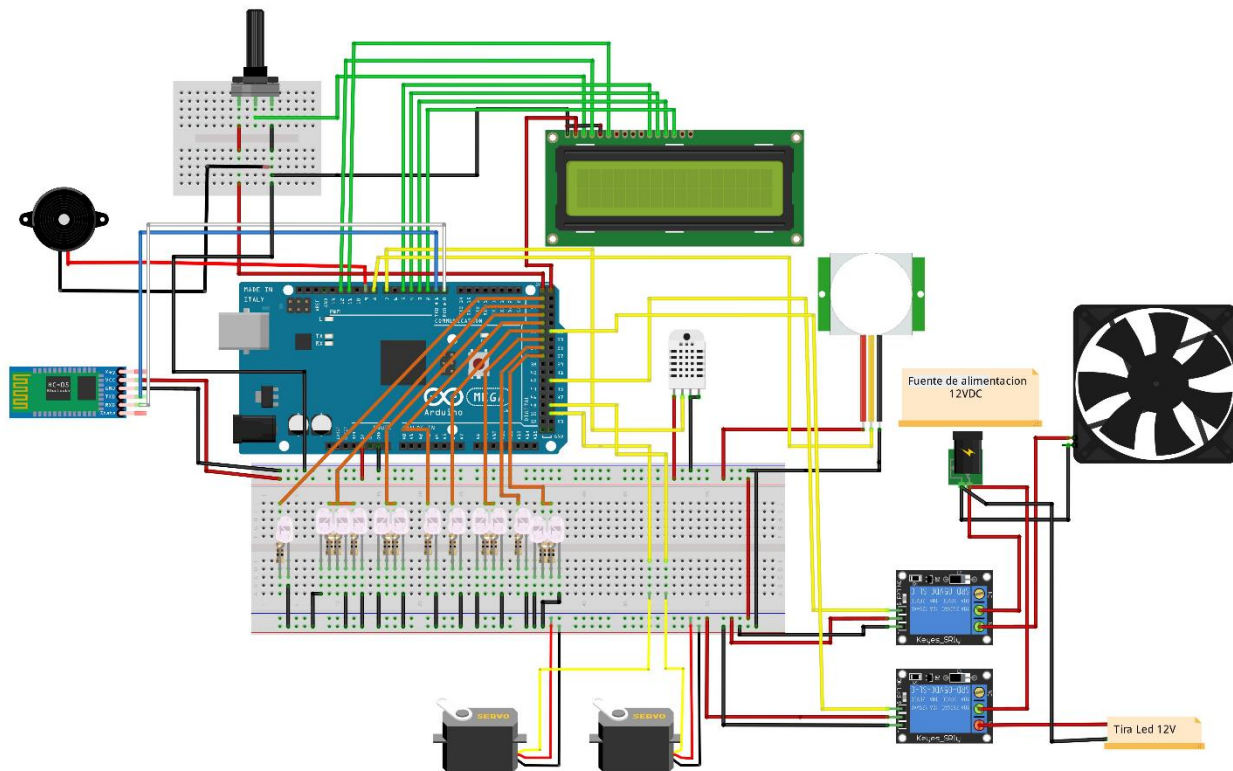


Fig. 7.1. Esquemático Proyecto Final

## 7.2. Implementación Maqueta

Con un diseño óptimo funcionando y con materiales listos para la construcción, se logra la implementación del proyecto hogar establecido utilizando como referencia un hogar real a escala. La implementación de la vivienda es el esqueleto de todo el proyecto y se demuestra en la Fig. 7.2.



Fig. 7.2. Maqueta Base Proyecto Hogar

Utilizando tornillos tradicionales para unir las paredes entre sí y silicona para los retoques o puntas cerradas. Además se atornilló a la base para dar estabilidad.

Luego se procedió a la instalación del Arduino y la luminaria más específicamente, para generar sentido y un orden al momento de la implementación.



### 7.3. Sistema de Luces

Para introducir luminaria en la maqueta se respetaron estándares básicos como ubicación y materiales necesarios. Se requirió la mayor similitud a un hogar real, por lo tanto, las luces están puestas estratégicamente en la maqueta de manera que no se contaminen visualmente las habitaciones. Además se procuró darle una forma de lámparas de pared que habitualmente hay en hogares.

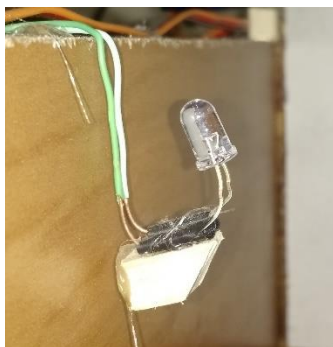


Fig. 7.3. Luces de Pared

Para verificar el funcionamiento de estas luces, bastaba con introducir el Arduino en la habitación especial, manipular las conexiones y alimentar. Como nuestro diseño cumple con los requisitos, se pudo iluminar la maqueta perfectamente, además reaccionan a la interacción con la aplicación.



Fig. 7.4. Maqueta Iluminada

#### 7.4. Verificación de la Comunicación

Con una maqueta lista para la manipulación, se necesita asegurar el perfecto funcionamiento del enlace bluetooth, por lo tanto, conectamos el módulo al Arduino MEGA, confirmamos comunicación y procedemos. Se necesitarán cables híbridos y el módulo bluetooth.

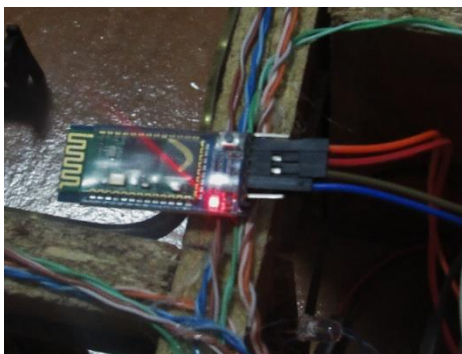


Fig. 7.5. Módulo Bluetooth activado

Una forma sencilla de comprobar la comunicación es utilizando la aplicación, ya que nos advierte si es que no hay dispositivos enlazados, de esta forma se debe modificar la conexión de cables o los códigos de programación.

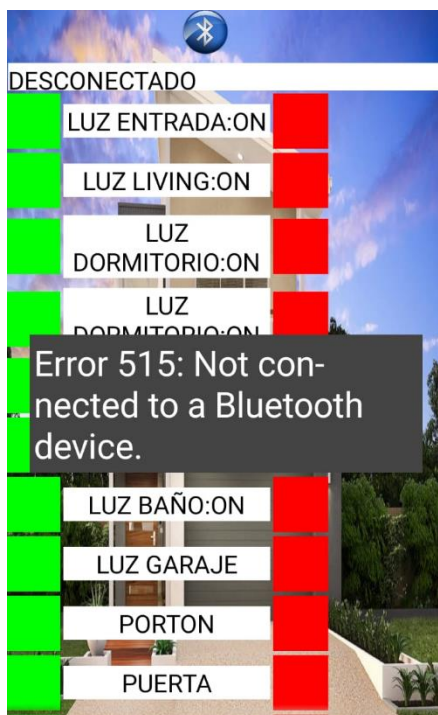


Fig. 7.6. Error Típico en la Comunicación

## 7.5. Sistema de Alarmas

Propiedad fundamental en la seguridad del sistema es la utilización de alarmas que advierten las intrusiones y nuestra maqueta está equipada para esto. Con una tira de leds adherida a las paredes del living-comedor se cuenta con la seguridad mínima que el sistema requiere.



Fig. 7.7. Tira de Leds en funcionamiento

Se logró el funcionamiento y detección de intrusión al pasar por el sensor de movimiento, que es el modo por el cual se gatilla esta advertencia. En conjunto con la tira de leds, instalamos el buffer que se activa de la misma manera. La desventaja del sistema de seguridad es su dependencia de a la corriente, cosa que en estos tiempos es fundamental un seguro o alerta, por esto, es recomendable en una replicación a escala real, tener algún dispositivo (UPS) que proporcione alimentación en caso de cortes de corriente o irrupciones al hogar.

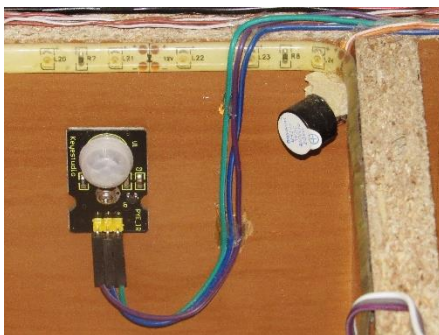


Fig. 7.8. Sensor de movimiento y buffer

## 7.6. Entradas On/Off

Como acceso principal al proyecto hogar se utilizaron materiales de manualidades que representarán la puerta principal de la casa. Esta está accionada por un servomotor y reacciona a la orden por parte de la aplicación móvil.



Fig. 7.9. Puerta principal de la casa

De la misma forma se tiene un portón que da al garaje de la casa, este portón funciona de la misma forma que la entrada principal, con un servomotor conectado al Arduino.



Fig. 7.10. Portón base de entrada a garaje

## 7.7 Ventilación y Temperatura

En función de lo amigable del proyecto en su funcionalidad e imagen, se le introdujo un sistema de ventilación que consiste en un ventilador de 12 [v] que cumple la función de aire acondicionado. Exitosamente cumple con el objetivo y responde a la comunicación vía bluetooth.

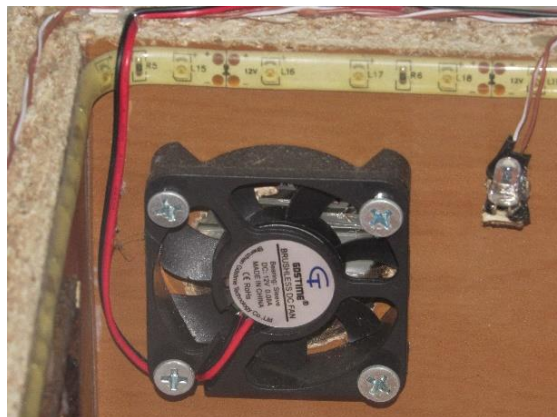


Fig. 7.11. Ventilador de 12 [V]

Secuencialmente, para denotar la temperatura se introdujo un sensor de temperatura conectado al microcontrolador, que a su vez indica la información de temperatura ambiente por medio de un display ubicado en el sector que representa la cocina del hogar. Además de la temperatura en grados celcius, denota la humedad presente en el ambiente, proporcionando una información más completa y de fácil acceso.



Fig. 7.12. Mensaje de bienvenida al encender el sistema

La ventaja de poseer un display informativo, aparte de leer cuando el sensor de temperatura está activo y cuando no, es que entrega los datos de las otras unidades de control, como por ejemplo, cuando se prende o apaga una luz, o también como cuando se abre o cierra o una puerta. Esto proporciona registros de lo que se está utilizando, lo que mejora el plan de energía que se consume en el hogar

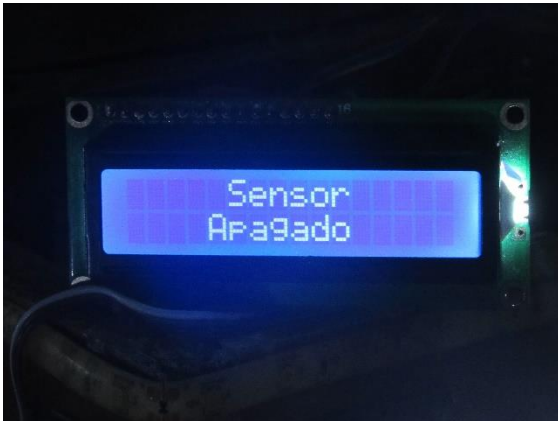


Fig. 7.13. Sensor de Temperatura sin activar



Fig. 7.14. Display con información del sensor de Temperatura

Tanto la información de temperatura ambiental, como la de datos de control pueden ser leídas en el display.



Fig. 7.15. Desactivación de las luces del living



Fig. 7.16. Activación de las luces del living vía aplicación móvil

## Conclusiones y Comentarios

Uno de los aspectos fundamentales de una aplicación domótica desarrollada con la plataforma Arduino es la accesibilidad en temas monetarios, es decir, su relación uso/costo. Es de bajo costo, con una alta versatilidad y con grandes expectativas de modificación. Una diferencia visible en comparación a otros modelos de la línea Arduino es su capacidad en cuanto a memoria, debido a esto y a la cantidad numerosa de controladores y actuadores presentes en la maqueta se prefirió usar el MEGA, ya que el formato UNO, NANO o DUE quedaban cortos en cuanto a memoria y capacidad de programación. Como desventaja, presenta la limitada cantidad de socks para pines y necesidad de ampliar funciones mediante otros dispositivos, como el hecho de la opción wifi, para lo que se necesita un shield conectado a la placa original. Con respecto a la comunicación bluetooth se debe denotar que el alcance máximo de interacción aplicación móvil-Arduino cubre los 10 metros como máximo, por lo tanto, dentro de una vivienda es perfectamente eficaz.

Debido a la cantidad de información que recibe el procesador tiende a producir un retraso al solicitar la información por la aplicación móvil. Esto ocurre principalmente con el sensor de movimiento y la activación de la alarma, el motivo es el peso del archivo, ya que supera los 25 Kb, lo que se demuestra en la activación de 1,5 (s) siendo lo normal un valor del orden de los microsegundos. Visualmente no es complicación en la ejecución del sistema, pero es una observación al momento de replicar la maqueta a una casa real. Precisamente esta replicación del control domótico a una vivienda real elevaría los precios ya que los materiales e instrumentos a usar serían de mayor tamaño y en mayor cantidad, pero el plano de conexiones eléctricas se mantendría.

Para la elaboración de los textos incluidos en el display, cuando se activa un control en el proyecto, se indica una información realizada mediante una programación usando como base un software dedicado incluido como parte del kit de compra del Arduino. Con este programa computacional, se definieron variables a utilizar y los diversos mensajes de activación y desactivación, también como la imagen de humedad y temperatura.

Arduino además de todas las ventajas ya mencionadas, la plataforma internacionalmente tiene una fama increíble, lo que proporciona soporte, ayuda e innovación en caso de problemas o deseos de expansión del sistema, es más, su compatibilidad con otras placas de diferentes distribuidores permiten optimización y autonomía garantizada, respetando el lenguaje de programación establecido. Existen páginas dedicadas, foros de personas conocedoras del tema en varios idiomas y proyectos propios de gente dispuesta a responder dudas y retroalimentación de conocimientos.

El lenguaje usado para la programación es el lenguaje C, que es un lenguaje bastante común y tradicional, fácil de aprender y lo mejor es que fácil de aplicar y sigue hoy en día marcando tendencias en aplicaciones a nivel mundial, ya sea empresariales, industriales o pymes comunitarias.

La necesidad de elaborar este proyecto, inicialmente surgió en el día a día de los autores. El trabajo personal sirvió como idea para su elaboración. El contexto y el carácter educativo se sobrepone al deseo de querer vender el sistema (opción que no se descarta) ya que a lo largo de la carrera no hay una asignatura que se enfoque en la innovación o implementación de estas placas o similares. Tras cada generación que entra a la carrera, siempre se observa el mismo déficit, las mismas falencias y con este proyecto se quiere suplir de cierta manera esa falta de actualidad, además de generar motivación propagando conocimiento de forma interactiva y útil. Se les favorecerá y facilitará el aprendizaje del lenguaje como también se les guiará en la estructura y conexiones. Como objetivo final está la opción de una seguidilla de ampliaciones que sometan al proyecto principal a una modificación que origine un idea nueva, con diferentes microcontroladores, que serán parte de una continuación de proyectos sucesivos.

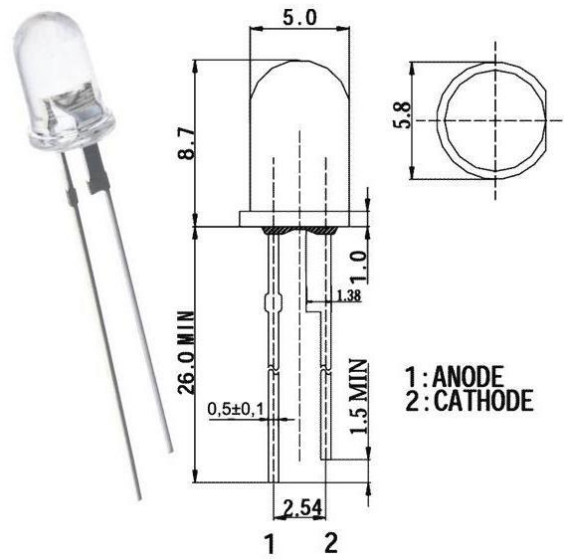


## Bibliografías

- [1] Arduino (sin fecha), revisado el 15 de agosto del 2018 en <https://www.arduino.cc>
- [2] Domótica: Introducción (sin fecha), revisado el 17 de agosto del 2018 en [http://www.profesormolina.com.ar/tecnologia/domotica/intro\\_domo.htm](http://www.profesormolina.com.ar/tecnologia/domotica/intro_domo.htm)
- [3] Sistema de comunicación digital, Julián Mejías, (2011), revisado el 18 de agosto del 2018 en <https://prezi.com/hpruydjf2o44/1-que-es-un-sistema-de-comunicacion-digital/>
- [4] Minera, Francisco Jose. “AJAX: WEB 2.0”, Editorial; Gradi, Año 2007.
- [5] Curso Práctico de Formación ARDUINO, Oscar Torrente Artero, Primera Edición (2013), Capítulos 2-3-4
- [6] Curso de programación en C, (sin fecha), revisado el 20 de agosto del 2018 en <https://www.aulafacil.com/cursos/programacion/lenguaje-de-programacion-c-t1454>
- [7] Rosado, Alfredo (sin fecha). Sistemas industriales distribuidos: Una filosofía de automatización. Recuperado el 25 de agosto del 2018, en [http://www.calsi.com/doc\\_tec/11.pdf](http://www.calsi.com/doc_tec/11.pdf)
- [8] Casas a escala, (sin fecha), revisado el 20 de agosto del 2018 en <http://construirtucasamini.blogspot.com/p/construccion-de-casitas.html>

## Anexos

### A1. Led 5mm Round White.



### Hoja de datos.

#### Absolute Maximum Ratings (Ta=25°C)

Parameter	Symbol	Rating	Unit
Continuous Forward Current	$I_F$	30	mA
Peak Forward Current(Duty /10 @ 1KHZ)	$I_{FP}$	100	mA
Reverse Voltage	$V_R$	5	V
Operating Temperature	$T_{opr}$	-40 ~ +85	°C
Storage Temperature	$T_{stg}$	-40 ~ +100	°C
Soldering Temperature (T=5 sec)	$T_{sol}$	260 ± 5	°C
Power Dissipation	$P_d$	100	mW
Zener Reverse Current	$I_z$	100	mA
Electrostatic Discharge	ESD	4K	V

## A2. Regulador Dc/Dc LM2596.



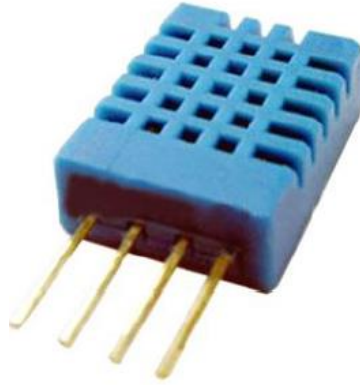
### Hoja de datos.

#### Absolute Maximum Ratings <sup>(1)(2)</sup>

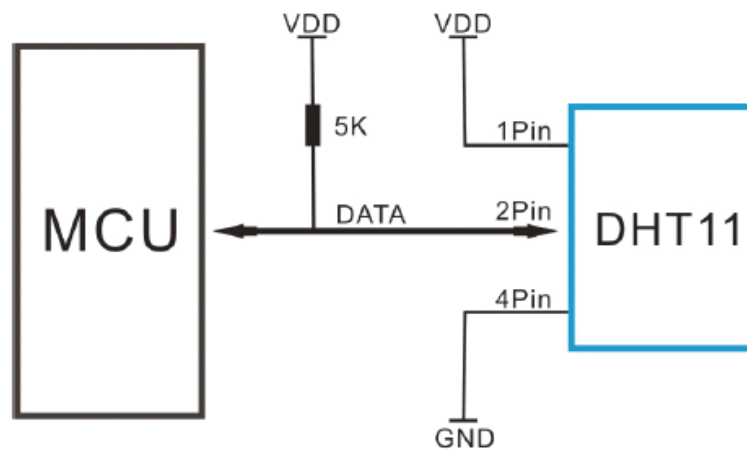
Maximum Supply Voltage	45V
$\overline{\text{ON}}$ /OFF Pin Input Voltage	$-0.3 \leq V \leq +25V$
Feedback Pin Voltage	$-0.3 \leq V \leq +25V$
Output Voltage to Ground (Steady State)	-1V
Power Dissipation	Internally limited
Storage Temperature Range	-65°C to +150°C
ESD Susceptibility	
Human Body Model <sup>(3)</sup>	2 kV
Lead Temperature	
DDPAK/TO-263 Package	
Vapor Phase (60 sec.)	+215°C
Infrared (10 sec.)	+245°C
TO-220 Package (Soldering, 10 sec.)	+260°C
Maximum Junction Temperature	+150°C

### A3. Sensor de Temperatura y Humedad DTH11.

#### Sensor DTH11



#### Conexión típica



**Hoja de datos.**

Parameters	Conditions	Minimum	Typical	Maximum
<b>Humidity</b>				
Resolution		1%RH	1%RH	1%RH
			8 Bit	
Repeatability			± 1%RH	
Accuracy	25 °C		± 4%RH	
	0-50 °C			± 5%RH
Interchangeability	Fully Interchangeable			
Measurement Range	0 °C	30%RH		90%RH
	25 °C	20%RH		90%RH
	50 °C	20%RH		80%RH
Response Time (Seconds)	1/e(63%)25 °C, 1m/s Air	6 S	10 S	15 S
Hysteresis			± 1%RH	
Long-Term Stability	Typical		± 1%RH/year	
Temperature				
Resolution		1 °C	1 °C	1 °C
		8 Bit	8 Bit	8 Bit
Repeatability			± 1 °C	
Accuracy		± 1 °C		± 2 °C
Measurement Range		0 °C		50 °C
Response Time (Seconds)	1/e(63%)	6 S		30 S

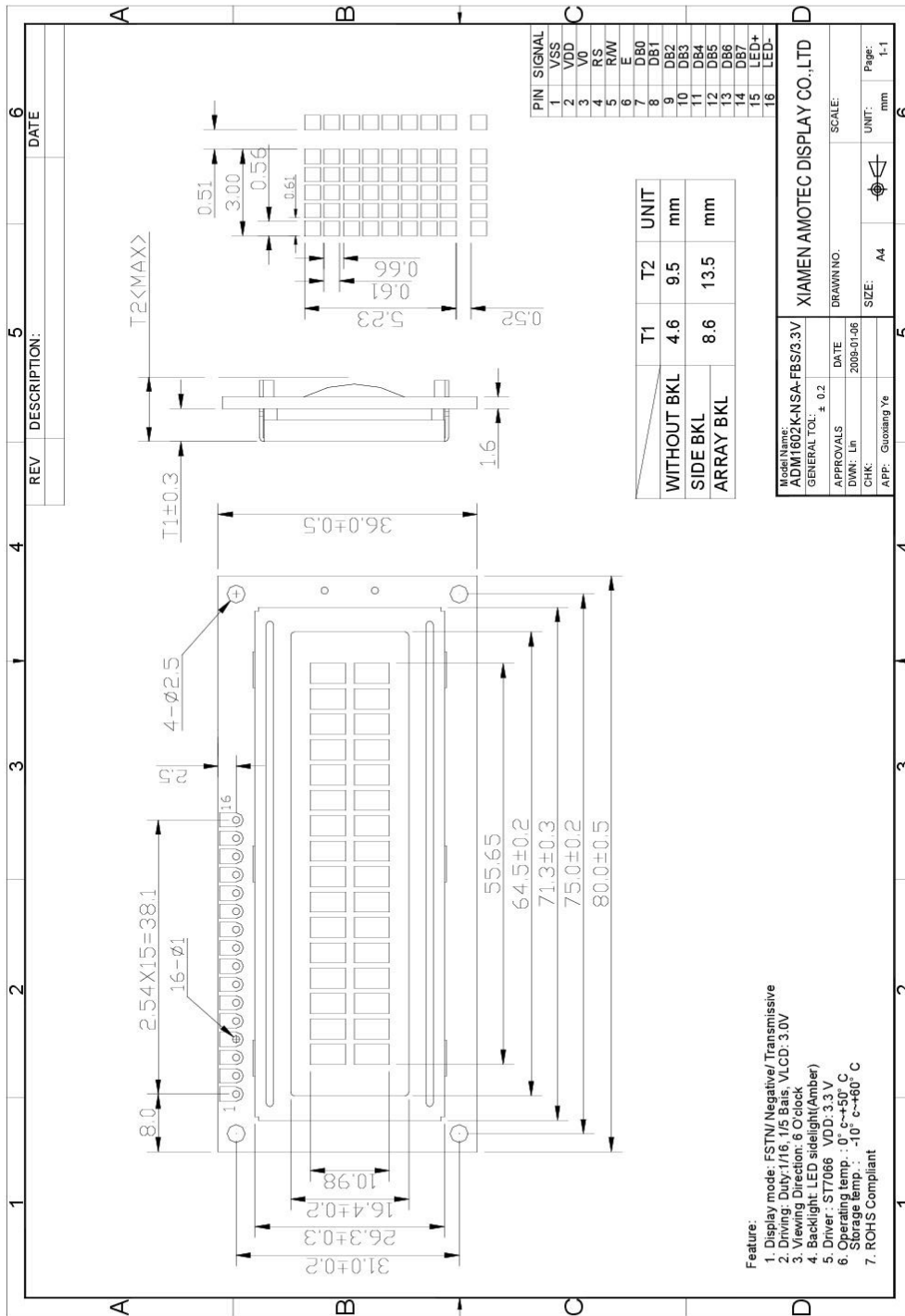
VDD=5V, T = 25 °C (unless otherwise stated)

	Conditions	Minimum	Typical	Maximum
Power Supply	DC	3V	5V	5.5V
Current Supply	Measuring	0.5mA		2.5mA
	Average	0.2mA		1mA
	Standby	100uA		150uA
Sampling period	Second	1		

Note: Sampling period at intervals should be no less than 1 second.

### A4. Display LCD 16x2

#### 3. Outline dimension

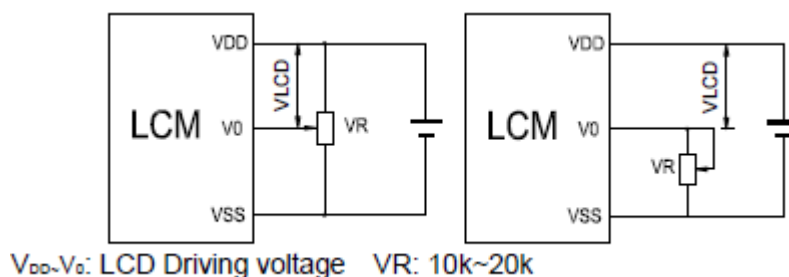


**Hoja de datos.**

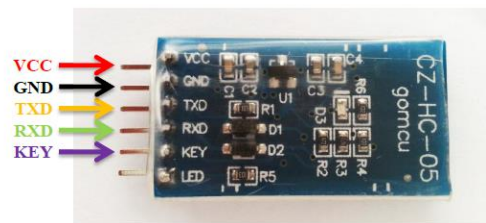
Item	Symbol	Standard			Unit
Power voltage	$V_{DD-V_{SS}}$	0	-	7.0	V
Input voltage	$V_{IN}$	VSS	-	VDD	
Operating temperature range	$V_{OP}$	0	-	+50	°C
Storage temperature range	$V_{ST}$	-10	-	+60	

Pin no.	Symbol	External connection	Function
1	VSS	Power supply	Signal ground for LCM
2	VDD		Power supply for logic for LCM
3	$V_0$		Contrast adjust
4	RS	MPU	Register select signal
5	R/W	MPU	Read/write select signal
6	E	MPU	Operation (data read/write) enable signal
7~10	DB0~DB3	MPU	Four low order bi-directional three-state data bus lines. Used for data transfer between the MPU and the LCM. These four are not used during 4-bit operation.
11~14	DB4~DB7	MPU	Four high order bi-directional three-state data bus lines. Used for data transfer between the MPU
15	LED+	LED BKL power supply	Power supply for BKL
16	LED-		Power supply for BKL

**Ajuste de contraste.**



## A5. Modulo Bluetooth HC-05



HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup.

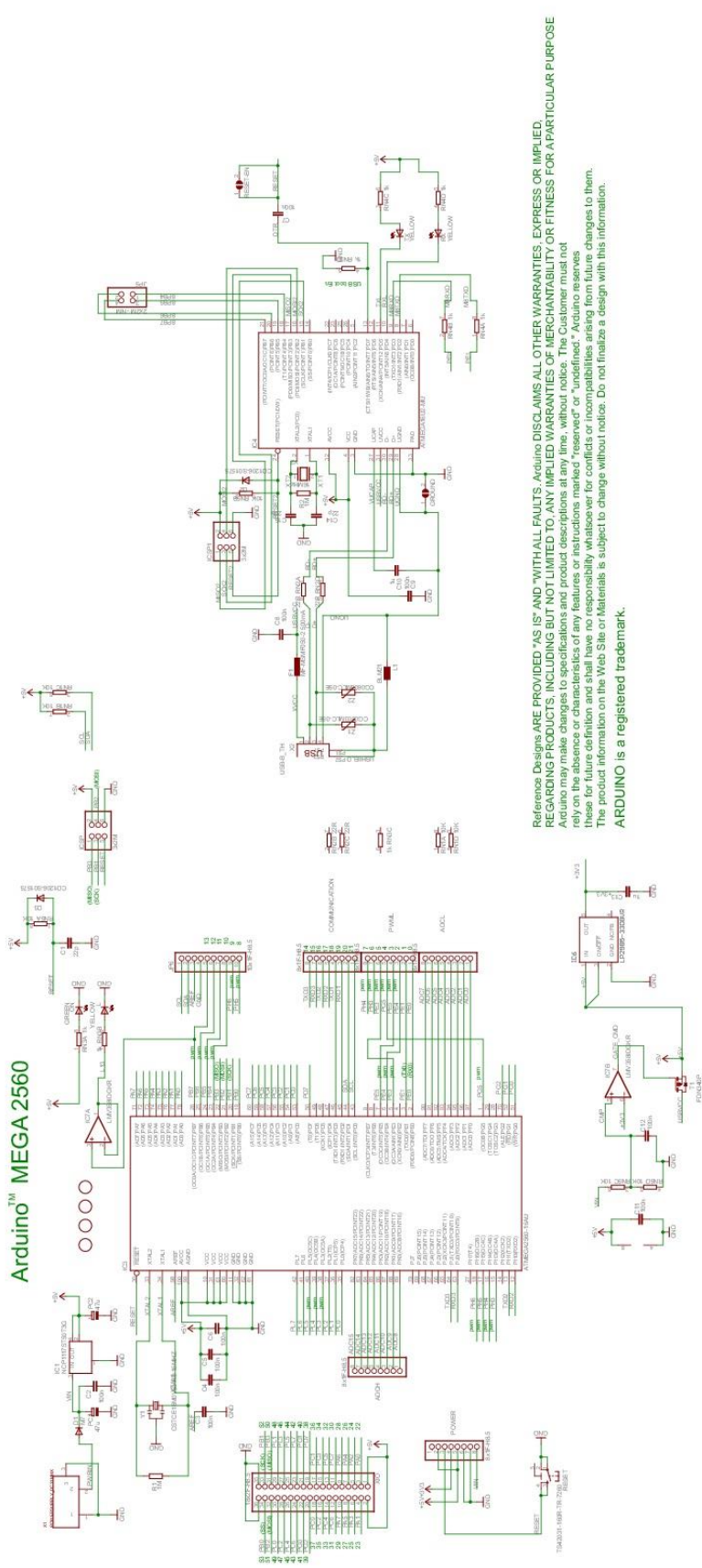
Serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH(Adaptive Frequency Hopping Feature). It has the footprint as small as 12.7mmx27mm. Hope it will simplify your overall design/development cycle.

### Hoja de datos.

Pin	Description	Function
VCC	+5V	Connect to +5V
GND	Ground	Connect to Ground
TXD	UART_TXD, Bluetooth serial signal sending PIN	Connect with the MCU's (Microcontroller and etc) RXD PIN.
RXD	UART_RXD, Bluetooth serial signal receiving PIN	Connect with the MCU's (Microcontroller and etc) TXD PIN.
KEY	Mode switch input	If it is input low level or connect to the air, the module is at paired or communication mode. If it's input high level, the module will enter to AT mode.



# A6.Arduino Mega 2560



Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS". Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Arduino does not warrant the accuracy, reliability, or completeness of the information provided. The information is provided for informational purposes only and should not be relied upon for any critical applications. The information is subject to change without notice. Do not finalize a design with this information.

**ARDUINO** is a registered trademark.

## A7. PIR sensor keystudio.



## Hoja de datos.

# keystudio

---

Input Voltage: 3.3 ~ 5V, 6V Maximum  
Working Current: 15uA  
Working Temperature: -20 ~ 85 °C  
Output Voltage: High 3V, low 0V  
Output Delay Time (High Level): About 2.3 to 3 Seconds  
Detection angle: 100 °  
Detection distance: 7 meters  
Output Indicator LED (When output HIGH, it will be ON)  
Pin limit current: 100mA  
Size: 30\*20mm  
Weight: 4g

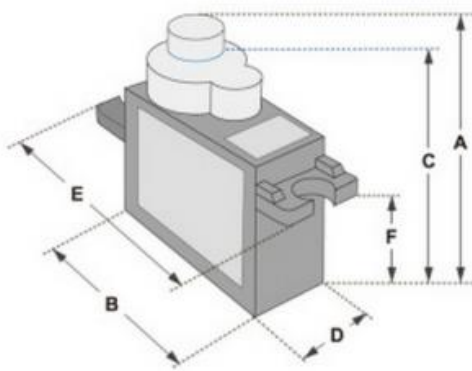
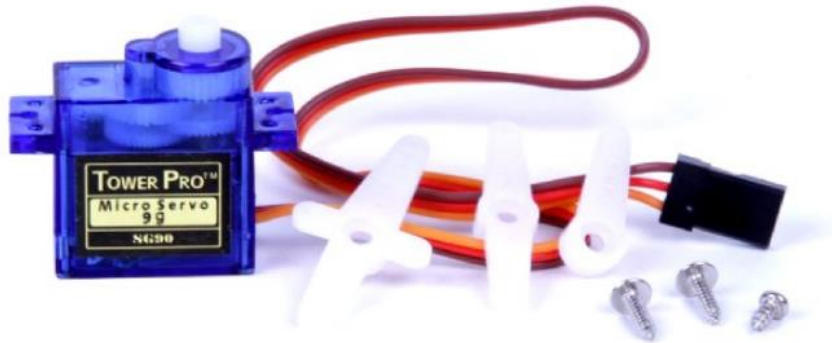
## A8. Buzzer activo.



### Hoja de datos.

- Buzzer o zumbador activo (Produce sonido al alimentarse)
- Rango del voltaje de alimentación: 1 V a 15 V
- Voltaje nominal de alimentación: 12 V
- Nivel de presión sonora mínima a 10 cm y 12 V: 85 dB
- Frecuencia de resonancia: 4000 ±500 Hz
- Corriente: 20 mA @ 12 V, 8 mA @ 5 V
- Temperatura de operación: -30 °C a +85 °C
- Material del encapsulado: Noryl
- Diámetro: 1.6 cm. Altura: 1.4 cm
- Peso: 2.5 g

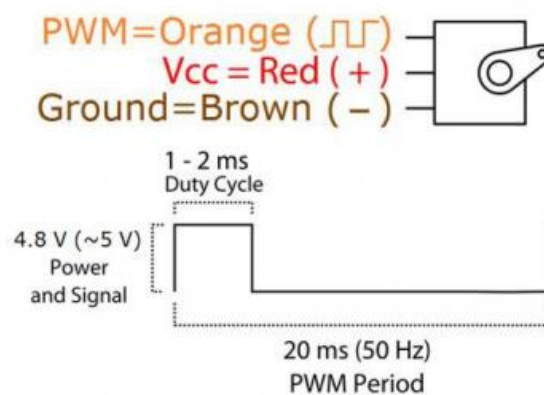
### A9. Servo motor 9G



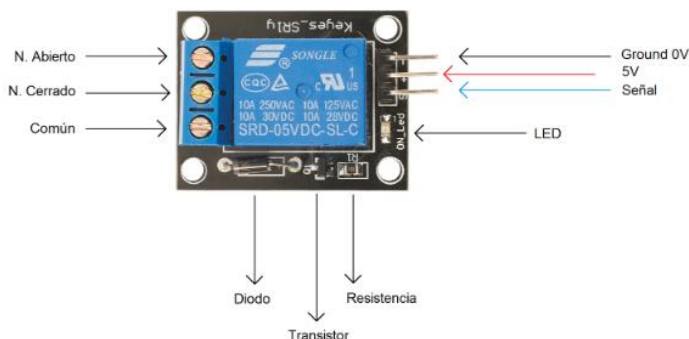
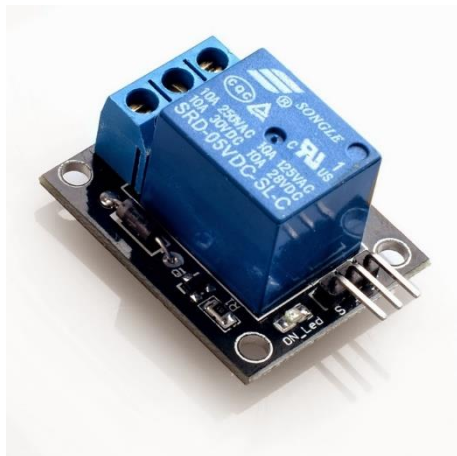
Dimensions & Specifications	
A (mm) :	32
B (mm) :	23
C (mm) :	28.5
D (mm) :	12
E (mm) :	32
F (mm) :	19.5
Speed (sec) :	0.1
Torque (kg-cm) :	2.5
Weight (g) :	14.7
Voltage :	4.8 - 6

### Hoja de datos.


Position "0" (1.5 ms pulse) is middle, "90" (~2ms pulse) is middle, is all the way to the right, "-90" (~1ms pulse) is all the way to the left.



**A10. Relé 5V DC – Keyes KY019.**



**Hoja de datos.**

	<b>RELAY ISO9002</b>	<b>SRD</b>
--	----------------------	------------



**1. MAIN FEATURES**

- Switching capacity available by 10A in spite of small size design for highdensity P.C. board mounting technique.
- UL,CUL,TUV recognized.
- Selection of plastic material for high temperature and better chemical solution performance.
- Sealed types available.
- Simple relay magnetic circuit to meet low cost of mass production.

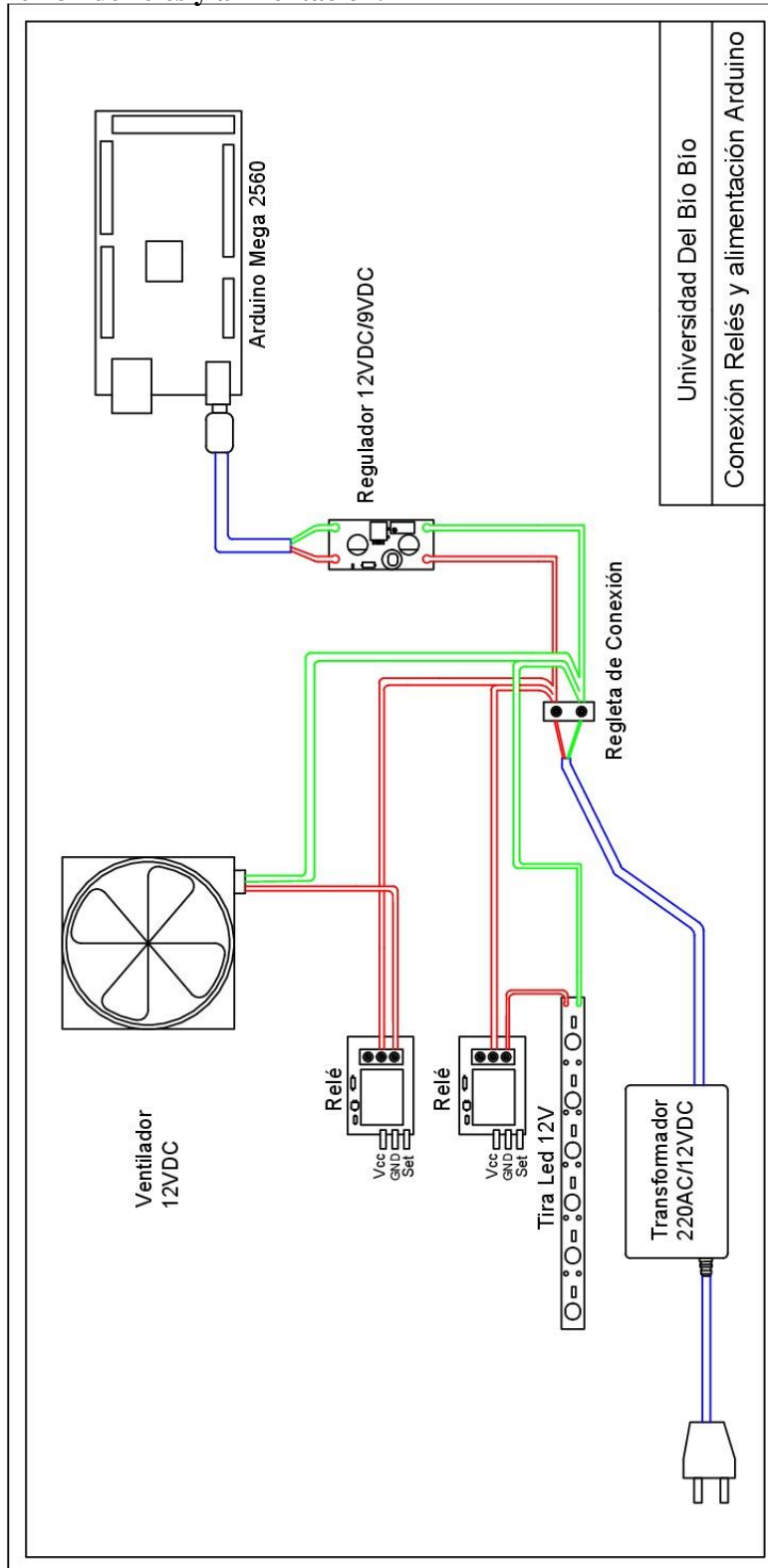
**2. APPLICATIONS**

- Domestic appliance, office machine, audio, equipment, automobile, etc.  
( Remote control TV receiver, monitor display, audio equipment high rushing current use application.)

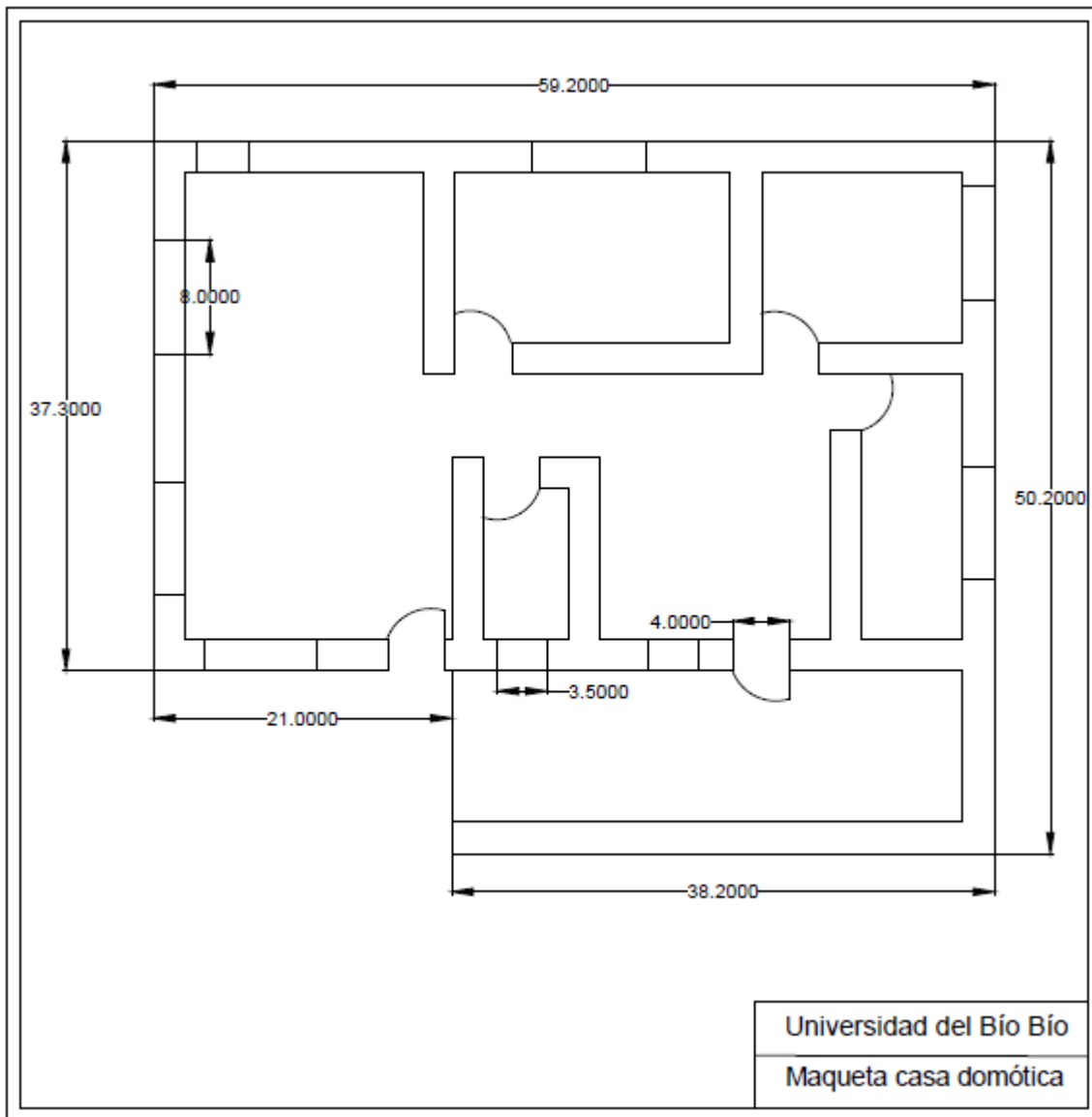
**3. ORDERING INFORMATION**

SRD	XX VDC	S	L	C
Model of relay	Nominal coil voltage	Structure	Coil sensitivity	Contact form
SRD	03、05、06、09、12、24、48VDC	S:Sealed type	L:0.36W	A:1 form A
		F:Flux free type	D:0.45W	B:1 form B C:1 form C

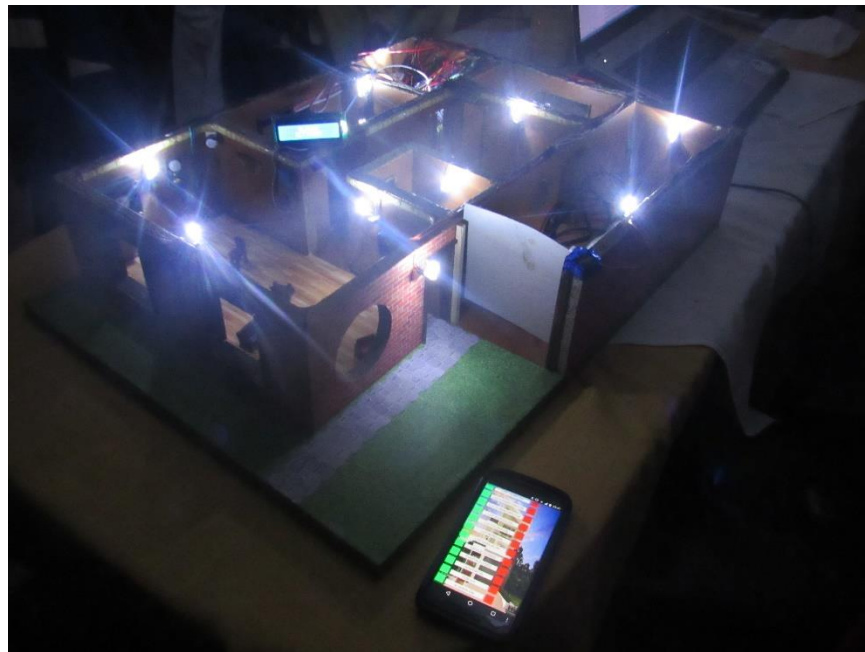
**A11. Plano conexión de relés y alimentación.**



**A12. Plano casa Domótica.**



**A13. Maqueta encendida**



**A14. Maqueta en estado de alarma**

