Arduino Básico, 1ª edición

Profesores: D. José Antonio Vacas Martínez
D. Juan Julián Merelo Guervós
D. Angel Pablo Hinojosa
D. José Antonio Serrano García

Fuente de alimentación de seguridad

Centro de Enseñanzas Virtuales de la Universidad de Granada.

Fundación General Universidad de Granada-Empresa.

Oficina de Software Libre.

Renato Luis Ramírez Rivero
Email: renatolrr@gmail.com
www.renatoramirez.com
Noviembre 2012

FUNDAMENTOS DEL PROYECTO.

Los ordenadores en estos momentos están considerados como parte primordial en nuestras vidas. No solo como elemento de trabajo, sino también como elemento de ocio pudiendo contener información privada y confidencial.

No solamente se pretende proteger nuestra información a nivel de software, sino que bloquear nuestro equipo a nivel de hardware, manteniendo segura la información contenida en el.

Todo el proyecto esta desarrollado utilizando software libre.

FUNDAMENTOS LEGALES DEL PROYECTO.

El artículo 18.4 de la Constitución Española establece que «la ley limitará el uso de la informática para garantizar el honor y la intimidad personal y familiar de los ciudadanos y el pleno ejercicio de sus derechos».

REAL DECRETO 994/1999, de 11 de junio, por el que se aprueba el Reglamento de medidas de seguridad de los ficheros automatizados que contengan datos de carácter personal.

Aquí te pongo unos artículos de esta ley que son interesantes:

Artículo 2. Definiciones.

A efectos de este Reglamento, se entenderá por:

1. Sistemas de información: conjunto de ficheros automatizados, programas, soportes y equipos empleados para el almacenamiento y tratamiento de datos de carácter personal.

2. Usuario: sujeto o proceso autorizado para acceder a datos o recursos.

3. Recurso: cualquier parte componente de un sistema de información.

4. Accesos autorizados: autorizaciones concedidas a un usuario para la utilización de los diversos recursos.

5. Identificación: procedimiento de reconocimiento de la identidad de un usuario.

6. Autenticación: procedimiento de comprobación de la identidad de un usuario.

7. Control de acceso: mecanismo que en función de la identificación ya autenticada permite acceder a datos o recursos.

8. Contraseña: información confidencial, frecuentemente constituida por una cadena de caracteres, que puede ser usada en la autenticación de un usuario.

9. Incidencia: cualquier anomalía que afecte o pudiera afectar a la seguridad de los datos.

10. Soporte: objeto físico susceptible de ser tratado en un sistema de información y sobre el cual se pueden grabar o recuperar datos.

11. Responsable de seguridad: persona o personas a las que el responsable del fichero ha asignado formalmente la función de coordinar y controlar las medidas de seguridad aplicables.

12. Copia del respaldo: copia de los datos de un fichero automatizado en un soporte que posibilite su recuperación.

Artículo 11. Identificación y autenticación.

1. El responsable del fichero se encargará de que exista una relación actualizada de usuarios que tengan acceso autorizado al sistema de información y de establecer procedimientos de identificación y autenticación para dicho acceso.

2. Cuando el mecanismo de autenticación se base en la existencia de contraseñas existirá un procedimiento de asignación, distribución y almacenamiento que garantice su confidencialidad e integridad.

3. Las contraseñas se cambiarán con la periodicidad que se determine en el documento de seguridad y mientras estén vigentes se almacenarán de forma ininteligible.

Artículo 12. Control de acceso.

1. Los usuarios tendrán acceso autorizado únicamente a aquellos datos y recursos que precisen para el desarrollo de sus funciones.

2. El responsable del fichero establecerá mecanismos para evitar que un usuario pueda acceder a datos o recursos con derechos distintos de los autorizados.

3. La relación de usuarios a la que se refiere el artículo 11.1 de este Reglamento contendrá el acceso autorizado para cada uno de ellos.

4. Exclusivamente el personal autorizado para ello en el documento de seguridad podrá conceder, alterar o anular el acceso autorizado sobre los datos y recursos, conforme a los criterios establecidos por el responsable del fichero.

Artículo 16. Responsable de seguridad.

El responsable del fichero designará uno o varios responsables de seguridad encargados de coordinar y controlar las medidas definidas en el documento de seguridad. En ningún caso esta designación supone una delegación de la responsabilidad que corresponde al responsable del fichero de acuerdo con este Reglamento.

Otros: LEY ORGÁNICA 5/1992

Se adjunta" Guía para el ciudadano" de la Agencia Española de Protección de Datos.

DESCRIPCION DEL PROYECTO

Se obtiene la imposibilidad de arrancar un equipo informático, sin conocer previamente la clave de acceso a este, ya que la fuente de alimentación consta de un circuito de corte de alimentación interno, con una clave que en un principio es pasado por cualquier medio, en nuestro caso físico (potenciómetro), pero que puede inclusive pasarse por Bluetooth con un teléfono con sistema operativo android, blackberry, etc. o con llave electrónica codificada.

Toda fuente a alimentación se activa conectando a tierra el cable verde (pin 14) del Main Power Connector) y tenemos que tener en cuenta que el cable violeta (pin 9) nos da una alimentación de 5v aunque la fuente no esté activada. Vamos a utilizar dicha alimentación para nuestro Arduino y utilizando un relé, conectaremos a tierra nuestro cable verde.

Nota: la idea es utilizar un Arduino Pro Mini 328 - 5V/16MHz

El botón de reset se soldará por la parte de atrás de la placa y se taladra la caja de la fuente, para su acceso con un puntero fino.

El espacio que disponemos para todo el circuito es de: 9x9x3 cm. De la misma fuente saldrá un cable que se conectará a la placa base, para pasar la información de la clave utilizando el programa de Perl.

En el frontal, tendremos un potenciómetro y un botón de envió de la lectura del potenciómetro. Nos dará el aspecto de la rueda de claves de una caja fuerte.

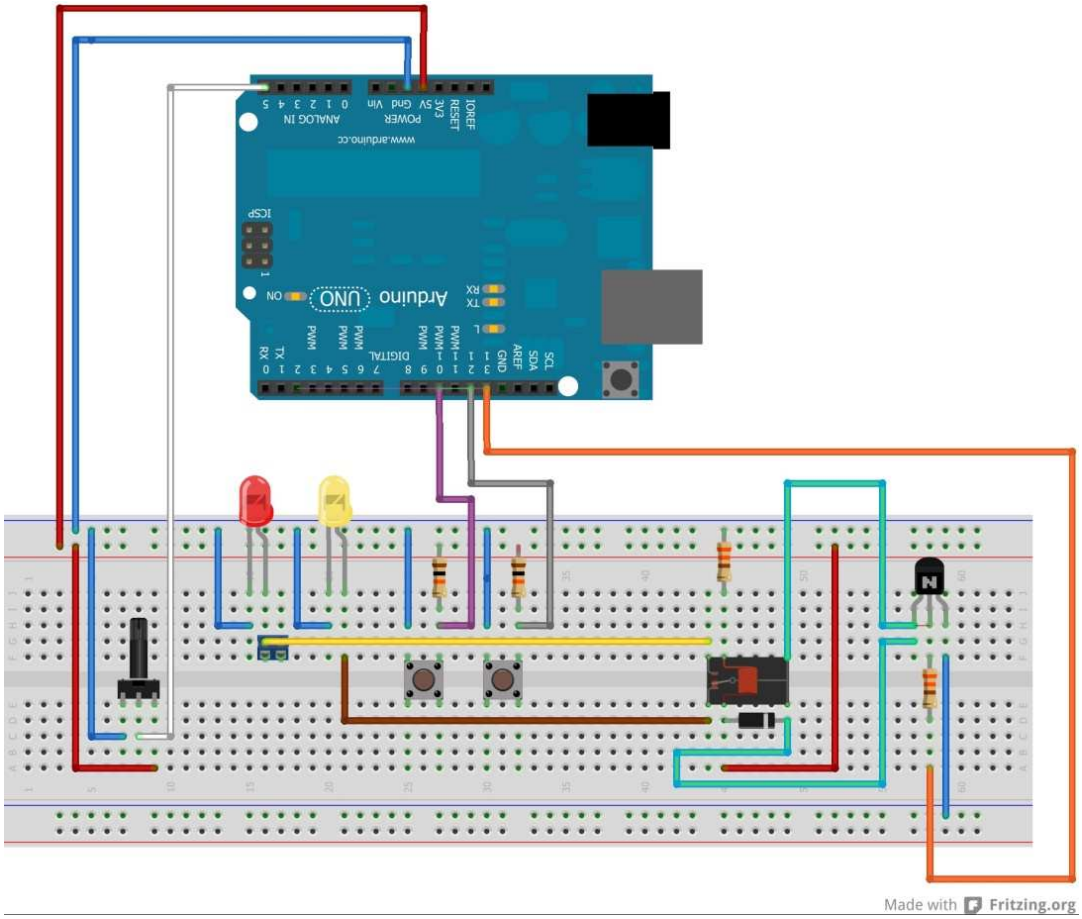
Se ha utilizado como ejemplo una placa Gigabyte ga-h61m-d2h. Se adjunta documentación.

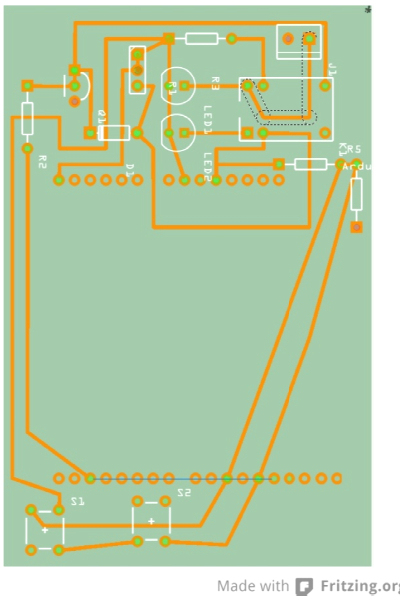
LISTADO DE COMPONENTES

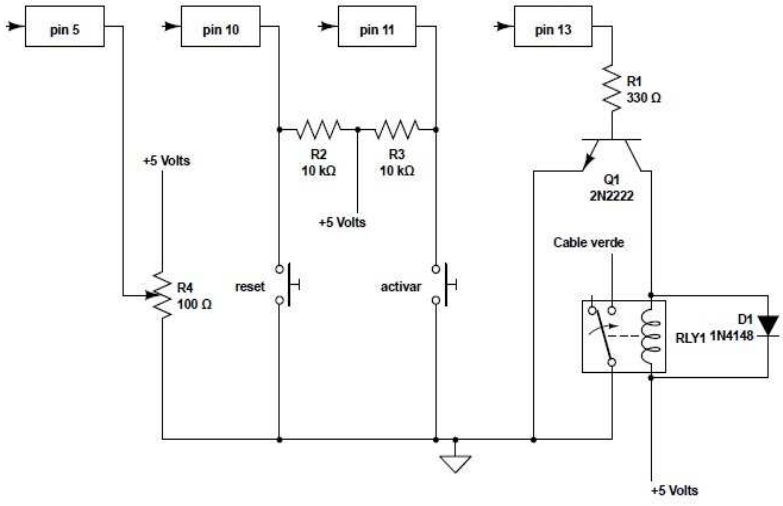
- Fuente alimentación atx
- Arduino
- Potenciómetro
- Botón de reset
- Botón de activación.
- Relé
- Transistor
- 2 resistencias de 10k
- 1 resistencia de 330
- Un diodo led
- Un transistor
- Un diodo
- Un conector

PROYECTO

- Esquema:







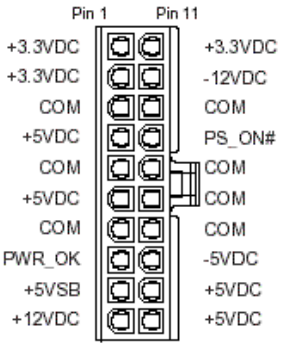
- Programación Arduino:

```
//Fuente de seguridad
//Renato Ramirez

#include <EEPROM.h>

int button1 = 10; // Boton de envio
int button2 = 12; // Boton de reset
int relayPin = 13;// Para activar transistor
int pot = 5; // Potenciometro de pass

void setup()
{
  pinMode(button1, INPUT); // envio como input
  pinMode(button2, INPUT); // reset como input
  pinMode(relayPin, OUTPUT);  // set pin as an output
  pinMode(pot, INPUT); // pass como input
  Serial.begin(9600); //serial
}

void loop()
{
  // reset
  int breset;
  breset = digitalRead(button2);
  if(breset == LOW){
    int ini=0;
    EEPROM.write(1,ini);
  }
  // lectura pass
  int pass;
  pass = analogRead(pot);
  // valor del pass
  int valor;
  valor = EEPROM.read(1);
  //cambio de valor
  if (Serial.available() > 0) {
     valor=Serial.read();
     EEPROM.write(1,valor);
     }
  // comprobar pass
  if(button1 == LOW){
    if((pass==valor)||(valor==0)){
      digitalWrite(relayPin,HIGH);
     }
  }
}
```

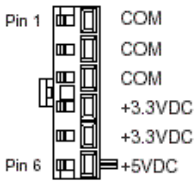
- Programacion clave en Perl:

```
use strict;
use warnings;

use Win32::SerialPort qw( :STAT 0.19 );

my $port = Win32::SerialPort->new('COM17');


if( ! defined($port) ) {
      die("Can't open COM17: $^E\n");
}

# $port->initialize();

$port->baudrate(9600);
$port->parity('none');
$port->databits(8);
$port->stopbits(1);
$port->write_settings();
$port->are_match("\n");

my $dur;
print "Valor de la clave? (1-10)\n";
$dur = <STDIN>;
$dur=ord($dur);
# Envio de Datos por el Puerto Serial
$port->write($dur);
print "Valor enviado ",$dur;
print "\n";
delay(0.1);
# Imprime el valor en memoria
my $char = $port->lookfor();
   if ($char) {
     print " Valor en memoria ",$char, "\n";
   }

$port->close();
 exit(0);

sub delay{ # Funcion para dar una pausa
          my $time=shift; # Variable que indica el tiempo(segundos), Con shift se recoge un
parametro a
          select(undef, undef, undef, $time); # Funcion Real que hace la pausa
}
```

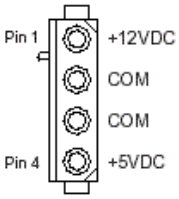
GALERÍA FOTOGRÁFICA.



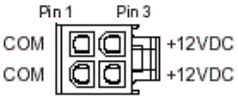
Main Power Connector



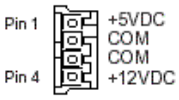
Aux Power Connector



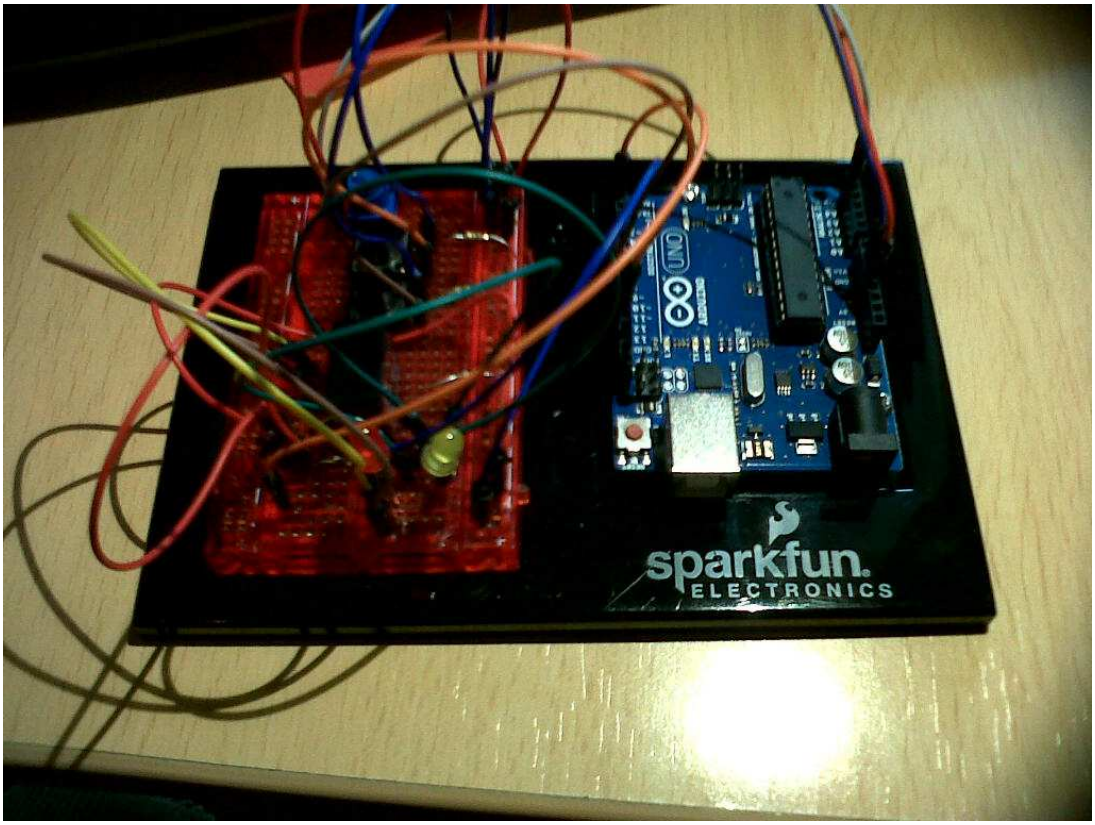
Peripheral Power Connector

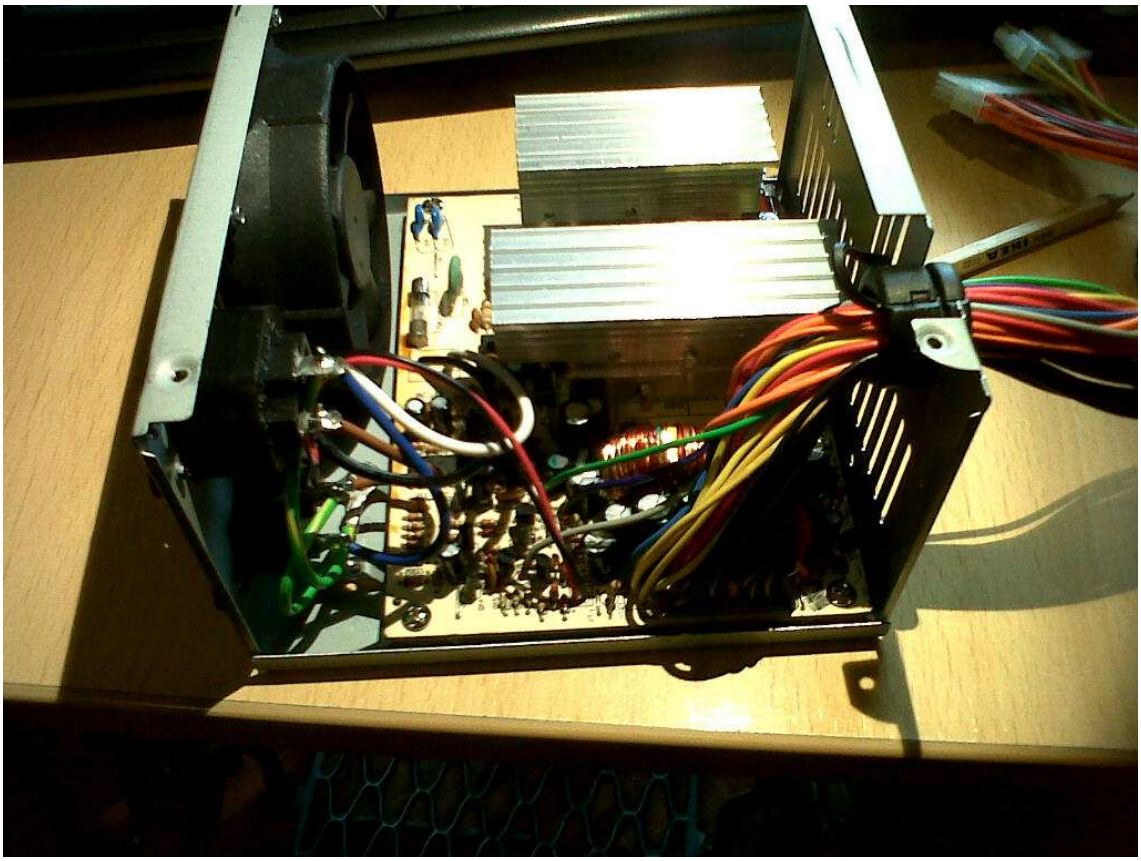


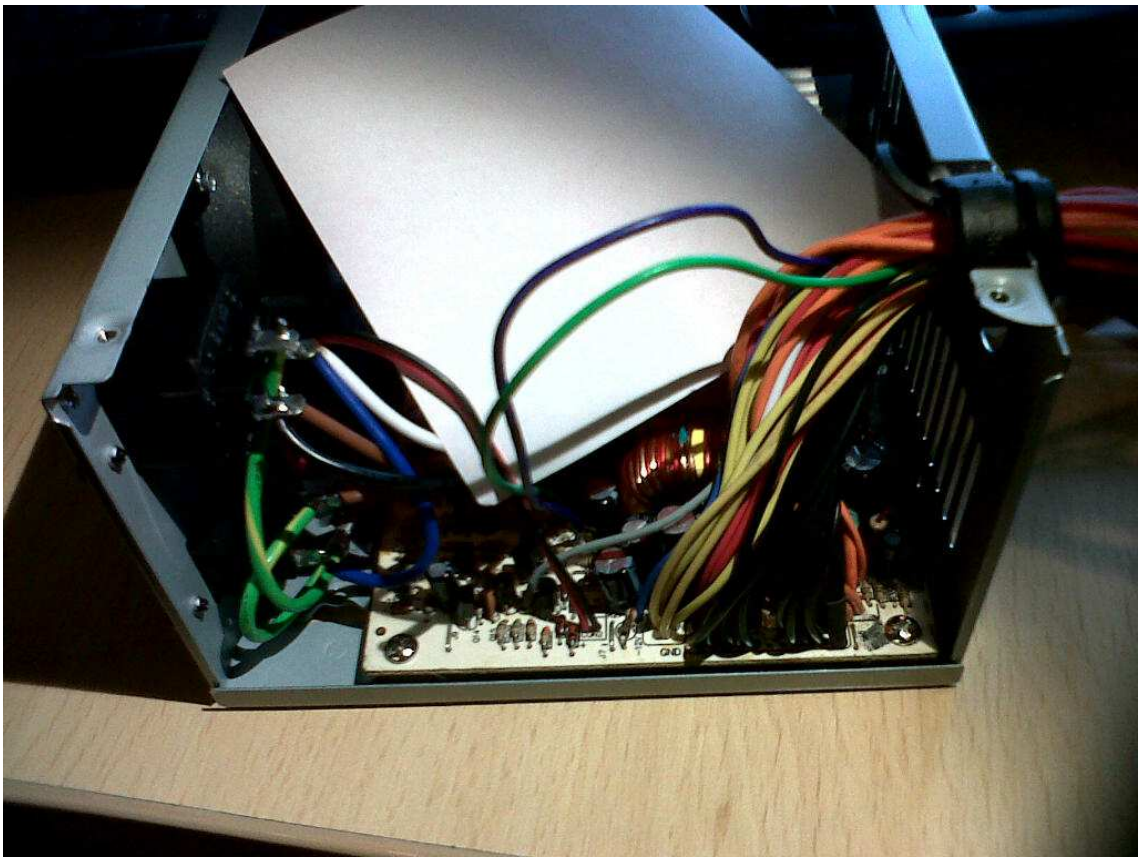
+12V Power Connector



Floppy Drive Power Connector

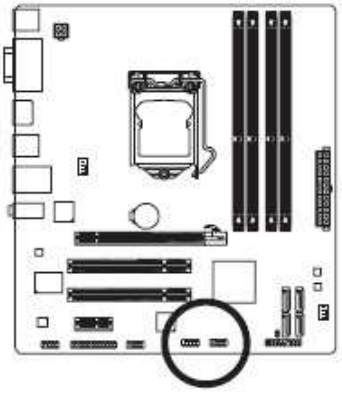






8) F_USB1/2 (USB 2.0/1.1 Headers)

The headers conform to USB 2.0/1.1 specification. Each USB header can provide two USB ports via an optional USB bracket. For purchasing the optional USB bracket, please contact the local dealer.



Pin No.	Definition
1	Power (5V)
2	Power (5V)
3	USB DX-
4	USB DY-
5	USB DX+
6	USB DY+
7	GND
8	GND
9	No Pin
10	NC

- Do not plug the IEEE 1394 bracket (2x5-pin) cable into the USB header.
- Prior to installing the USB bracket, be sure to turn off your computer and unplug the power cord from the power outlet to prevent damage to the USB bracket.

BIBLIOGRAFÍA

- Arduino Programing Notebook, Brian W. Evans, august 2007
- Material del curso: Arduino Básico de Centro de Enseñanzas Virtuales de la Universidad de Granada.
- Sik Guide, your guide to the Sparkfun Inventor´s Kit for Arduino.
- http://www.clubplaneador.com/datos_articulos/fuente_12v.htm
- Material del curso: Programación en Perl de Centro de Enseñanzas Virtuales de la Universidad de Granada.
- http://webdelcire.com/wordpress/archives/1045

AMPLIACIÓN DEL PROYECTO.

A) Utilizando un dispositivo android con Bluetooth.

Tendremos un menú inicial que nos mostrará los dispositivos Bluetooth disponibles y con un botón lo que haremos en enviar una señal al pin 13 de nuestro Arduino que nos activara el relé que conmuta el cable 13 con tierra de nuestra fuente de alimentación.

Código Arduino:

```
boolean estado;

void setup()
{
  Serial.begin(9600);
  pinMode(13, OUTPUT);
  digitalWrite(13, LOW);
  randomSeed(analogRead(0));
  estado = false;
}

void loop()
{
    delay(1000);
    Serial.write(random(40));
    while(Serial.available() > 0)
    {
      Serial.read();
      estado = !estado;
      digitalWrite(13, estado);
    }
}
```

Código Android:

```
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import java.util.ArrayList;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.lang.reflect.Method;

private static final int REQUEST_ENABLE_BT = 3;
ArrayList dispositivos;
BluetoothAdapter adaptador;
BluetoothDevice dispositivo;
BluetoothSocket socket;
InputStream ins;
OutputStream ons;
boolean registrado = false;
PFont f1;
PFont f2;
```

```
int estado;
String error;
byte valor;
//////////////////////////////////////////////////////////////////////
BroadcastReceiver receptor = new BroadcastReceiver()
{
    public void onReceive(Context context, Intent intent)
    {
        println("onReceive");
        String accion = intent.getAction();

        if (BluetoothDevice.ACTION_FOUND.equals(accion))
        {
            BluetoothDevice dispositivo =
intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            println(dispositivo.getName() + " " +
dispositivo.getAddress());
            dispositivos.add(dispositivo);
        }
        else if
(BluetoothAdapter.ACTION_DISCOVERY_STARTED.equals(accion))
        {
          estado = 0;
          println("Empieza búsqueda");
        }
        else if
(BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(accion))
        {
          estado = 1;
          println("Termina búsqueda");
        }

    }
};

//////////////////////////////////////////////////////////////////////
void setup() {
  //size(320,480);
  frameRate(25);
  f1 = createFont("Arial",20,true);
  f2 = createFont("Arial",15,true);
  stroke(255);
}
//////////////////////////////////////////////////////////////////////
void draw() {
  switch(estado)
  {
    case 0:
      listaDispositivos("BUSCANDO DISPOSITIVOS", color(255, 0, 0));
      break;
    case 1:
      listaDispositivos("ELIJA DISPOSITIVO", color(0, 255, 0));
      break;
    case 2:
      conectaDispositivo();
      break;
    case 3:
      muestraDatos();
      break;
    case 4:
      muestraError();
```

```
      break;
  }
}
//////////////////////////////////////////////////////////////////////
void onStart()
{
  super.onStart();
  println("onStart");
  adaptador = BluetoothAdapter.getDefaultAdapter();
  if (adaptador != null)
  {
    if (!adaptador.isEnabled())
    {
        Intent enableBtIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
    }
    else
    {
      empieza();
    }
  }
}
//////////////////////////////////////////////////////////////////////
void onStop()
{
  println("onStop");
  /*
  if(registrado)
  {
    unregisterReceiver(receptor);
  }
  */

  if(socket != null)
  {
    try
    {
      socket.close();
    }
    catch(IOException ex)
    {
      println(ex);
    }
  }
  super.onStop();
}
//////////////////////////////////////////////////////////////////////
void onActivityResult (int requestCode, int resultCode, Intent data)
{
  println("onActivityResult");
  if(resultCode == RESULT_OK)
  {
    println("RESULT_OK");
    empieza();
  }
  else
  {
    println("RESULT_CANCELED");
    estado = 4;
    error = "No se ha activado el bluetooth";
```

```
  }
}
////////////////////////////////////////////////////////////////////
void mouseReleased()
{
  switch(estado)
  {
    case 0:
      /*
      if(registrado)
      {
        adaptador.cancelDiscovery();
      }
      */
      break;
    case 1:
      compruebaEleccion();
      break;
    case 3:
      compruebaBoton();
      break;
  }
}
////////////////////////////////////////////////////////////////////
void empieza()
{
    dispositivos = new ArrayList();
    /*
    registerReceiver(receptor, new
IntentFilter(BluetoothDevice.ACTION_FOUND));
    registerReceiver(receptor, new
IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_STARTED));
    registerReceiver(receptor, new
IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_FINISHED));
    registrado = true;
    adaptador.startDiscovery();
    */
    for (BluetoothDevice dispositivo : adaptador.getBondedDevices())
    {
        dispositivos.add(dispositivo);
    }
    estado = 1;
}
////////////////////////////////////////////////////////////////////
void listaDispositivos(String texto, color c)
{
  background(0);
  textFont(f1);
  fill(c);
  text(texto,0, 20);
  if(dispositivos != null)
  {
    for(int indice = 0; indice < dispositivos.size(); indice++)
    {
      BluetoothDevice dispositivo = (BluetoothDevice)
dispositivos.get(indice);
      fill(255,255,0);
      int posicion = 50 + (indice * 55);
      if(dispositivo.getName() != null)
      {
        text(dispositivo.getName(),0, posicion);
```

```
      }
      fill(180,180,255);
      text(dispositivo.getAddress(),0, posicion + 20);
      fill(255);
      line(0, posicion + 30, 319, posicion + 30);
    }
  }
}
//////////////////////////////////////////////////////////////////
void compruebaEleccion()
{
  int elegido = (mouseY - 50) / 55;
  if(elegido < dispositivos.size())
  {
    dispositivo = (BluetoothDevice) dispositivos.get(elegido);
    println(dispositivo.getName());
    estado = 2;
  }
}
//////////////////////////////////////////////////////////////////
void conectaDispositivo()
{
  try
  {
    socket =
dispositivo.createRfcommSocketToServiceRecord(UUID.fromString("0000110
1-0000-1000-8000-00805F9B34FB"));
    /*
      Method m =
dispositivo.getClass().getMethod("createRfcommSocket", new Class[] {
int.class });
      socket = (BluetoothSocket) m.invoke(dispositivo, 1);
    */
    socket.connect();
    ins = socket.getInputStream();
    ons = socket.getOutputStream();
    estado = 3;
  }
  catch(Exception ex)
  {
    estado = 4;
    error = ex.toString();
    println(error);
  }
}
//////////////////////////////////////////////////////////////////
void muestraDatos()
{
  try
  {
    while(ins.available() > 0)
    {
      valor = (byte)ins.read();
    }
  }
  catch(Exception ex)
  {
    estado = 4;
    error = ex.toString();
    println(error);
  }
```

```
  background(0);
  fill(255);
  text(valor, width / 2, height / 2);
  stroke(255, 255, 0);
  fill(255, 0, 0);
  rect(120, 400, 80, 40);
  fill(255, 255, 0);
  text("Botón", 135, 425);
}
//////////////////////////////////////////////////////////////////////
void compruebaBoton()
{
  if(mouseX > 120 && mouseX < 200 && mouseY > 400 && mouseY < 440)
  {
    try
    {
        ons.write(0);
    }
    catch(Exception ex)
    {
      estado = 4;
      error = ex.toString();
      println(error);
    }
  }
}
//////////////////////////////////////////////////////////////////////
void muestraError()
{
  background(255, 0, 0);
  fill(255, 255, 0);
  textFont(f2);
  textAlign(CENTER);
  translate(width / 2, height / 2);
  rotate(3 * PI / 2);
  text(error, 0, 0);
}
//////////////////////////////////////////////////////////////////////
```

B) Utilizando una sd (llave)

Solamente se tiene que conectar un cable de las rx y tx de nuestro Arduino al usb de la caja del ordenador, y se realiza una lectura de un fichero que tengamos en dicha SD.

Como leer y escribir en una sd con Arduino:

http://arduino.cc/en/Tutorial/ReadWrite