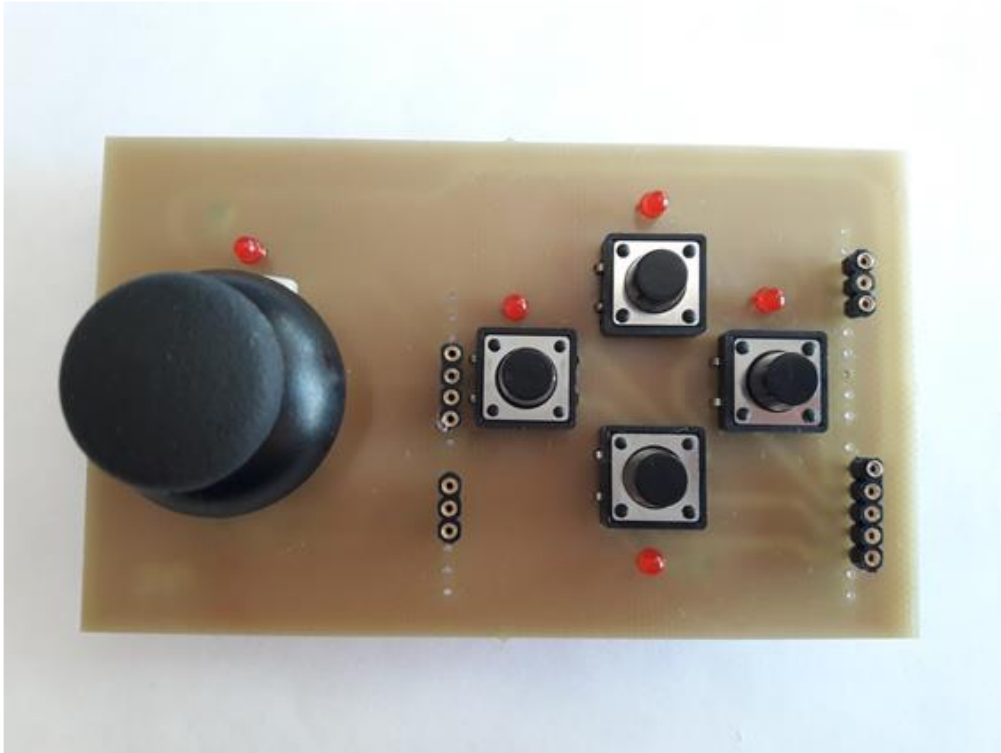


MEMORIA DE PRÁCTICA

DISEÑO DE UNA PCB

(MANDO DE JUEGOS)



Nombre: Francisco Núñez Cid

Correo: francisco.nunez.cid@gmail.com

Curso: 2016 - 2017

Asignatura: LDH - 4º IC

ÍNDICE

• Introducción	2
• Objetivos	2
• Fases de proceso de una PCB	
○ Fase de Diseño	3
▪ ¿Qué es Eagle?	3
▪ Utilización del Software	3
▪ ¿Qué es KiCad?	15
▪ Utilización del Software	15
▪ Comparativa Eagle y KiCad	24
○ Fase de Fabricación	25
○ Fase de Ensamblaje	26
○ Fase de Test	30
• Pruebas con Arduino	31
○ PCB como mando de juegos	31
○ PCB como ratón	40
• Conclusión	43
• Glosario	44
• Bibliografía	46

Introducción

Durante el desarrollo de todas las prácticas de laboratorio del segundo bloque, he ido tomando una serie de notas de lo que he ido realizado, los problemas que me he ido encontrando a la hora de hacer la PCB (esquemático, layout, ensamblaje de componentes y verificación), la solución que he obtenido de los problemas encontrados y los diferentes resultados que voy concluyendo mostrándolo con imágenes.

Se pretende diseñar una placa de expansión para la placa de desarrollo Arduino que funcione de manera similar a un mando de consola: con joystick y pulsadores.

Objetivos

Los principales propósitos de las prácticas de diseño de PCB son los siguientes:

- Instalar y configurar el software suministrado para realizar los diseños de la PCB (se utilizarán Eagle y KiCad, ambos son software libres).
- Realizar los diseños “Esquemáticos” y “Layout” de la PCB a través de los software: Eagle y KiCad.
- Fabricación de la PCB sin errores en el diseño.
- Ensamblaje de los componentes en la PCB (resistencias, leds, botones...).
- Desarrollo de software sobre Arduino verificando la placa de expansión desarrollada.
- Desarrollo de software sobre PC utilizando el mando.

Fases de proceso de una PCB

El proceso de implementación en una PCB de un sistema electrónico consta de las siguientes partes: Fase de Diseño, Fase de Fabricación, Fase de Ensamblaje y Fase de Test o Verificación.

- **Fase de Diseño**

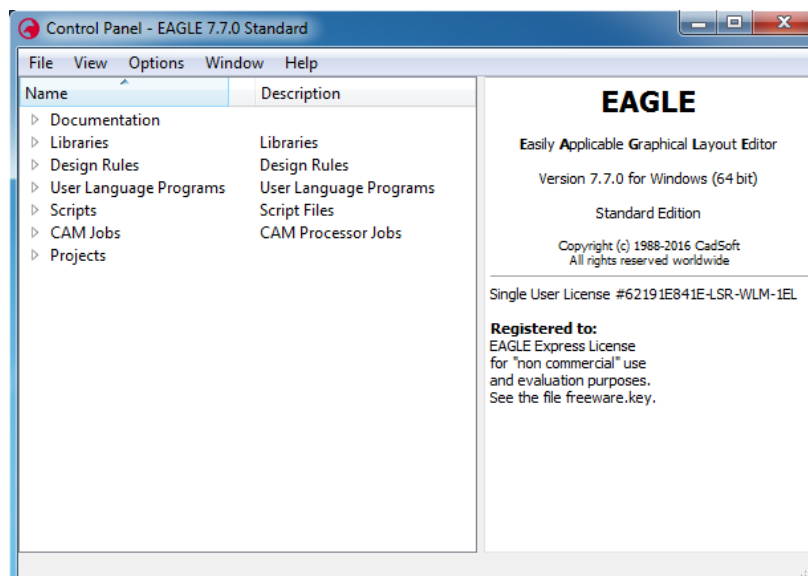
En esta fase se va a crear el “Layout” o “Artwork” de la PCB del SE. Para realizar los diseños (Esquemático y layout) se han utilizado los siguientes software: Eagle y KiCad.

- **¿Qué es Eagle?**

Es una herramienta (software) que se utiliza para el diseño de circuitos electrónicos y diagramas de la PCB. Además, muchas versiones de este programa tienen una licencia Freeware y una gran cantidad de bibliotecas de componentes alrededor de la red.

- **Utilización del Software**

1. Primero, nos descargamos e instalamos Eagle.
2. Ejecutamos el programa y se nos debería de mostrar la siguiente imagen.



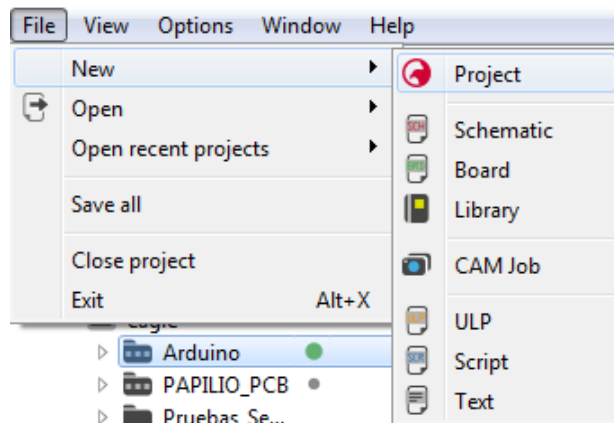
En el panel superior, observamos las siguientes pestañas:

- **File:** Sirve para crear o abrir un proyecto, esquemático, board, librería, etc. y guardarlos.
- **View:** Sirve para extender el modo de visualización de la ventana, refrescar (F5), etc.
- **Options:** Sirve para los directorios, uso de la interfaz, etc.
- **Window:** Sirve para ver la ventana del proyecto creado.
- **Help:** Sirve para dar ayuda sobre Eagle en general, el contexto, sobre el panel de control, licencias, chequeos, etc.

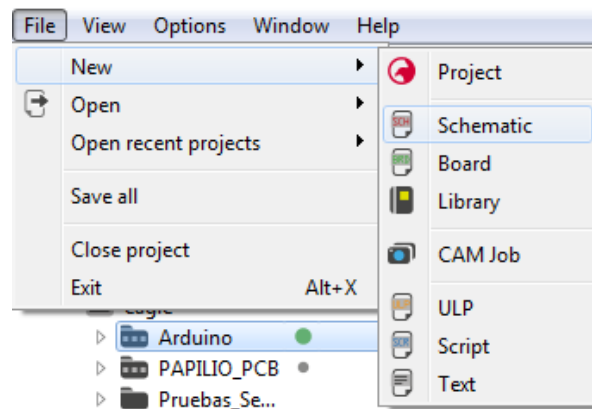
En el panel central, observamos las siguientes pestañas:

- **Documentation:** Sirve de ayuda con diferente documentación sobre Eagle y ejemplos del uso.
- **Libraries:** Son las bibliotecas que contienen los diferentes componentes electrónicos. Además se pueden añadir o crear.
- **Design Rules:** Muestran las reglas que tienen los diseños a través de Eurocircuit, Multi-CB, etc.
- **User Language Programs:** Muestra los programas de lenguaje usado.
- **Scripts:** Almacena los “.scr” de Eagle.
- **CAM Jobs:** Almacena los trabajos “.cam”. Sirve para generar los gerbers una vez hecho el diseño de la PCB.
- **Projects:** Almacena los proyectos creados y ejemplos que ya tiene integrado Eagle.

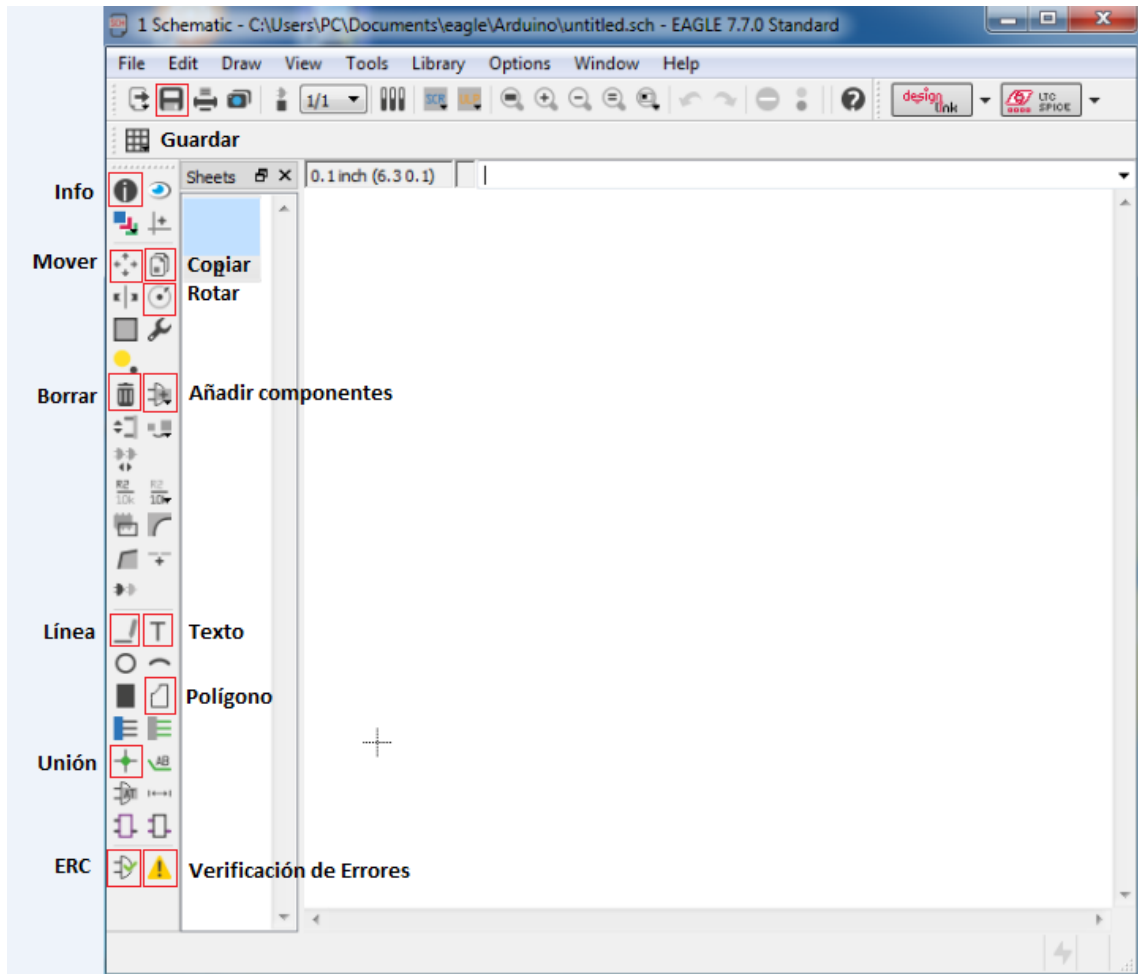
3. Me creo un proyecto, que lo he llamado “**Arduino**” y se crea dentro de la carpeta Eagle. **File -> New -> Project**.



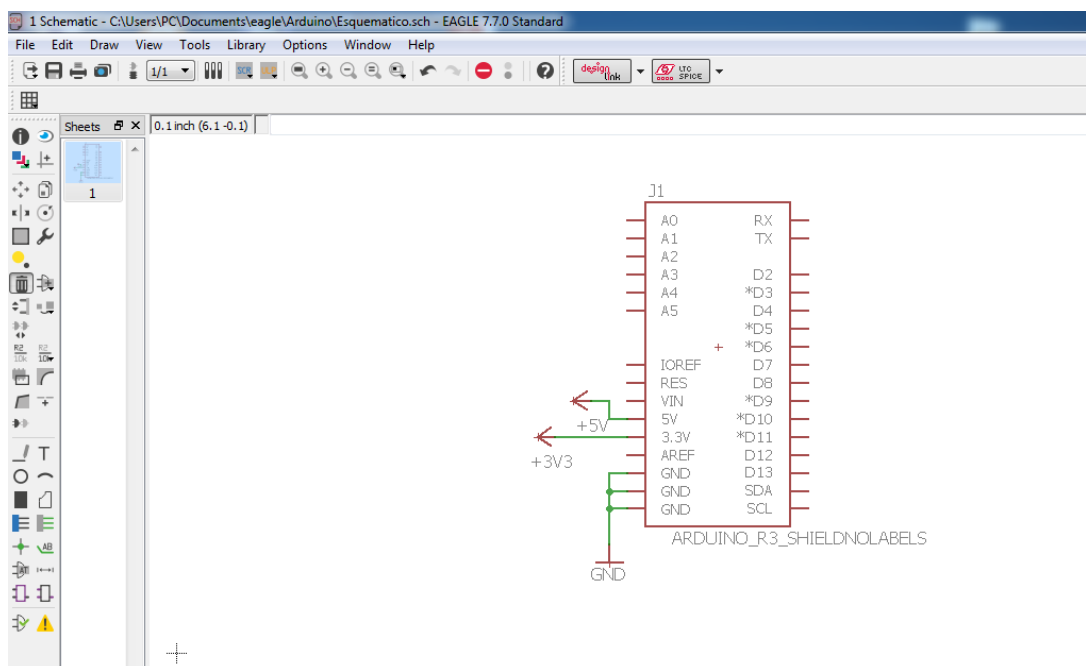
4. Descargar las librerías de Sparkfun (ver enlace en Bibliografía) para algunos componentes (joystick, botón y Arduino) del diseño a realizar de la PCB.
5. Una vez creado el proyecto de inicio y añadida las librerías necesarias para los componentes, paso a crearme el esquemático. **File -> New -> Schematic**.



6. Se debe de abrir una nueva ventana en blanco. Los elementos señalados son los más importantes y los que he utilizado para realizar el esquemático de la PCB.



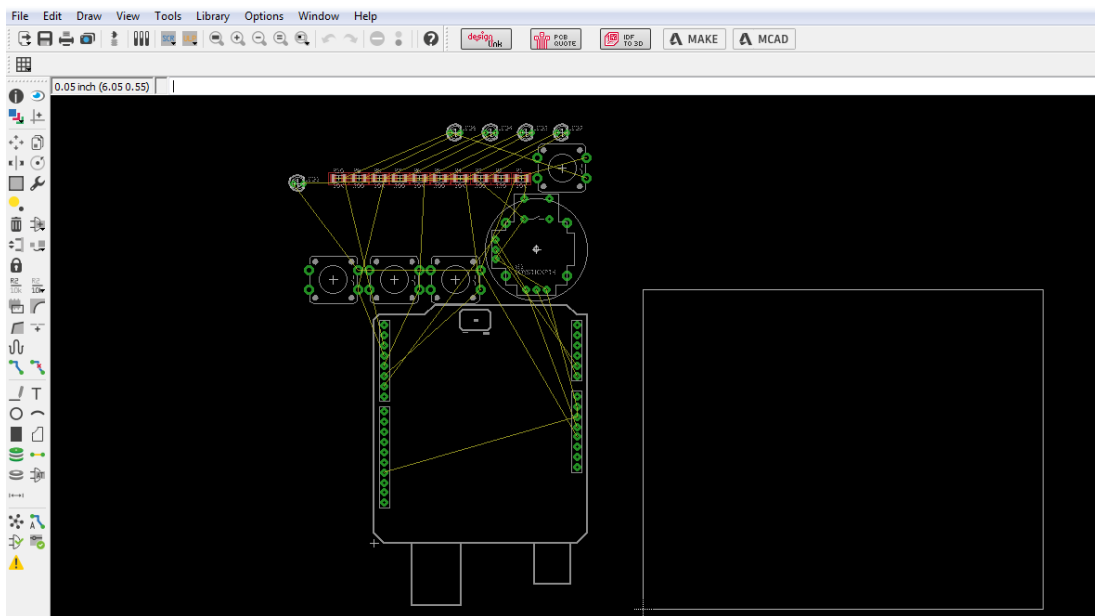
7. Empezamos a añadir componentes y he comenzado añadiendo el Arduino, señalando las entradas de 5V, 3.3V y las salidas GND.



10. Las conexiones que van a un mismo cable las he unido con el botón (**junction**) y después de realizar el circuito completo lo he verificado, pulsando el botón (**ERC o Verificación de Errores**).

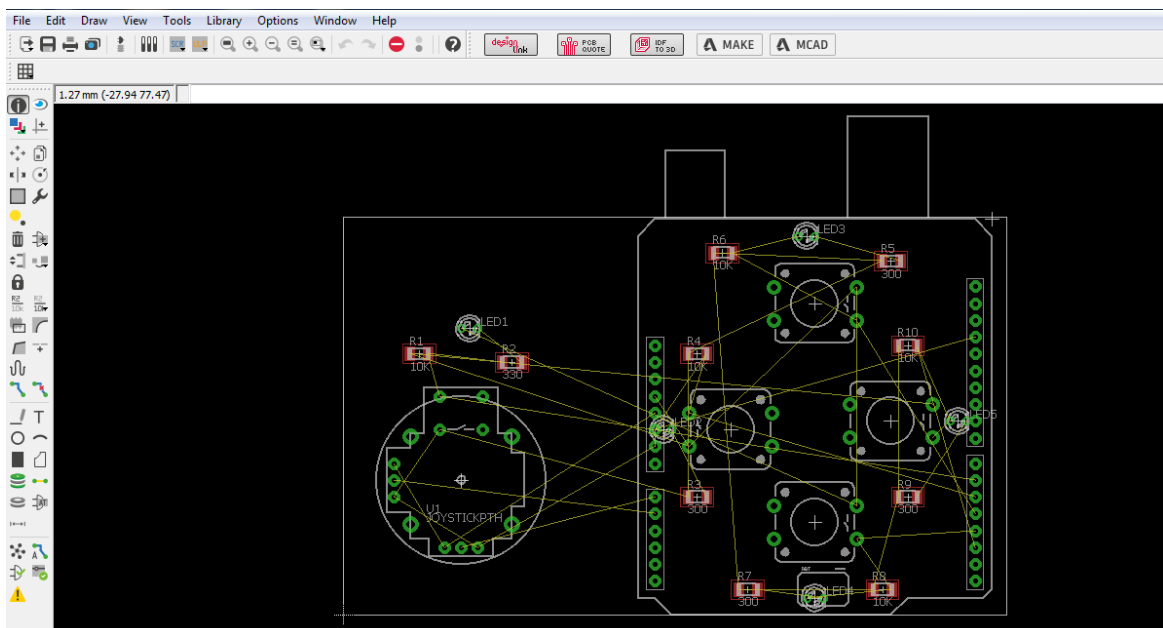
11. Una vez hecho el “Esquemático”, pasamos a realizar el “Layout”.

File -> Switch to board. Se nos abre una nueva ventana con los componentes de la PCB que hemos colocados en el Esquemático.

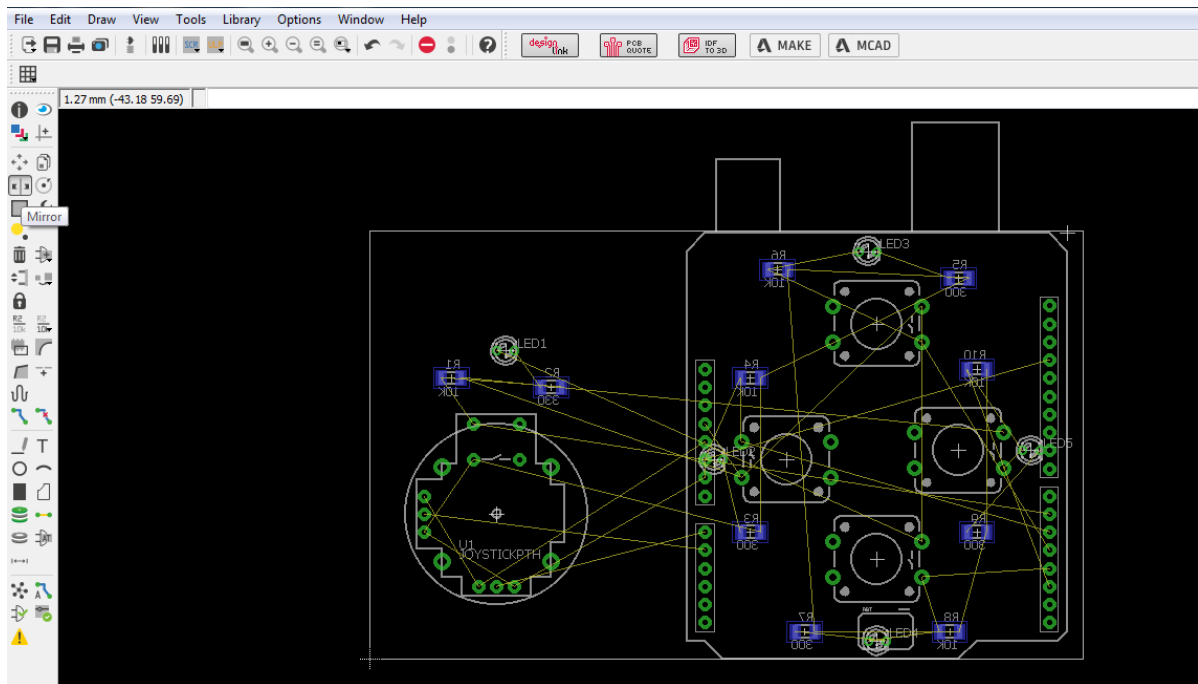


12. Cambio el tamaño de la PCB y lo pongo en milímetros, a 10 x 6 cm.

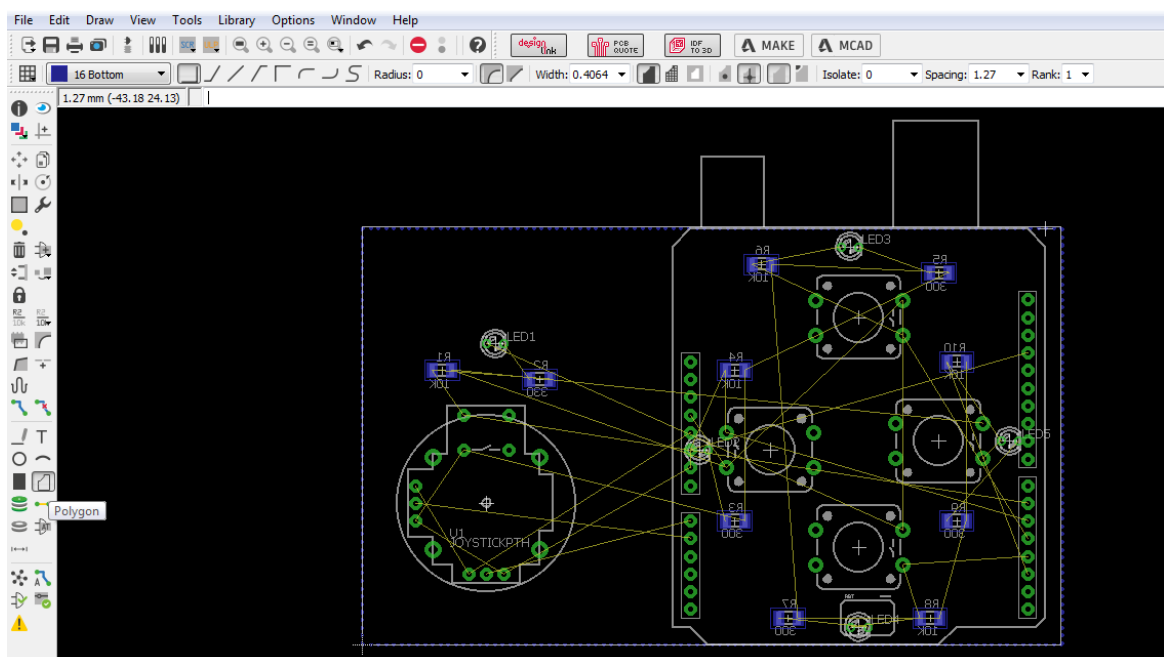
13. A continuación, paso a colocar los componentes en la PCB.



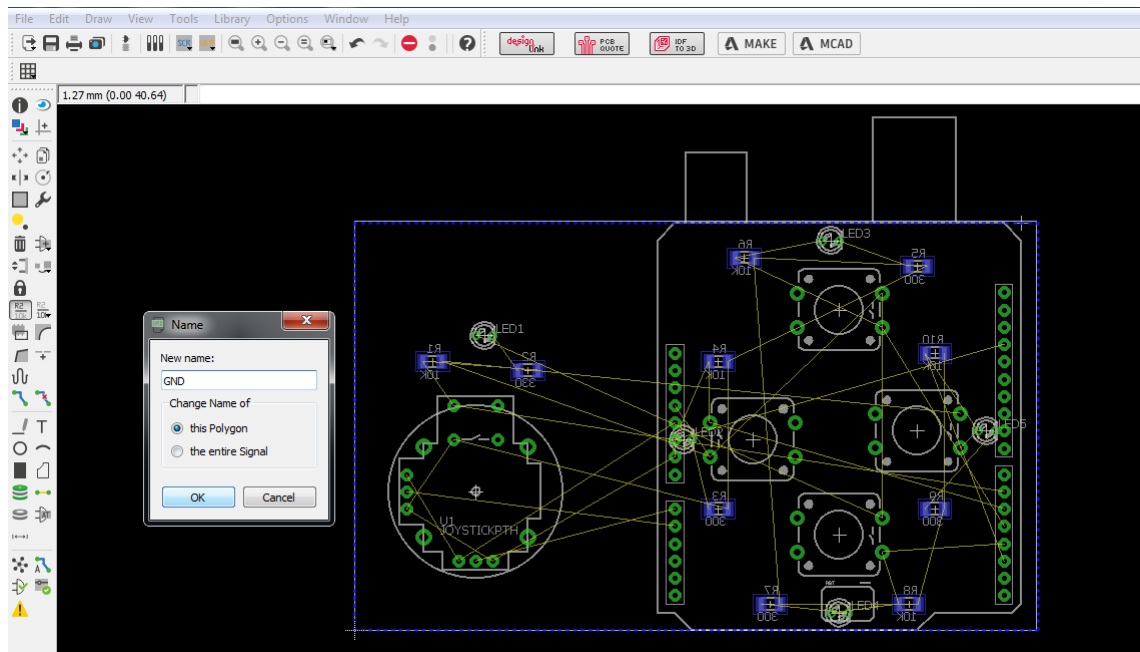
14. Pulso sobre el botón “**Mirror**” y le doy la vuelta a las resistencias de 10K y de 300 Ohm.



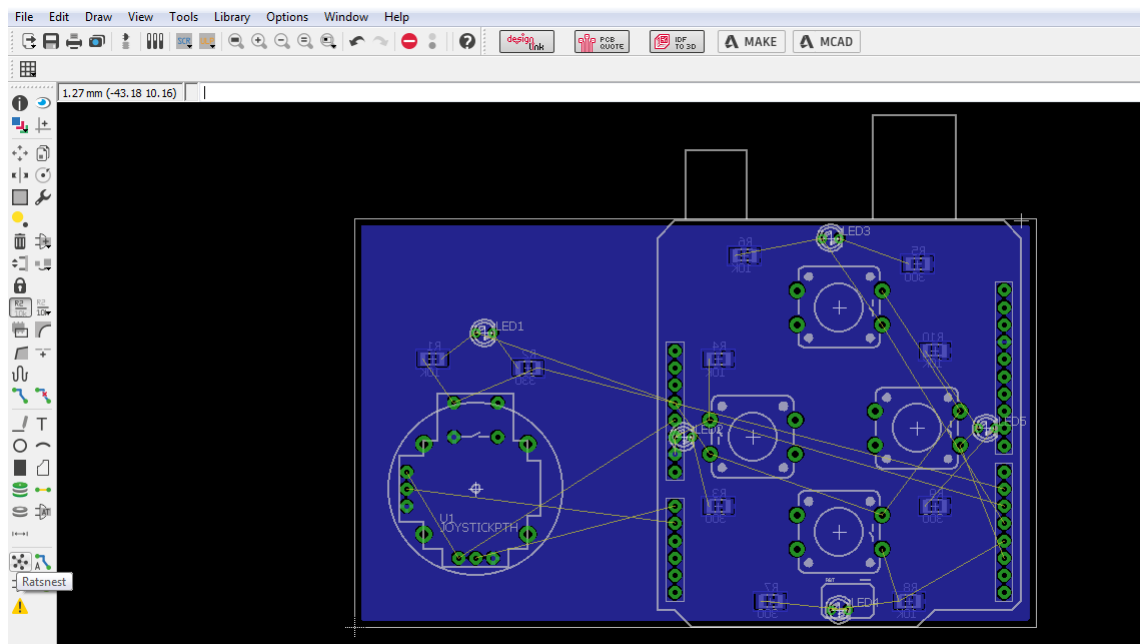
15. Ahora paso a crearme el “Plano de tierra (masa)”, pulso sobre “**Polygon**” y remarco toda la PCB.



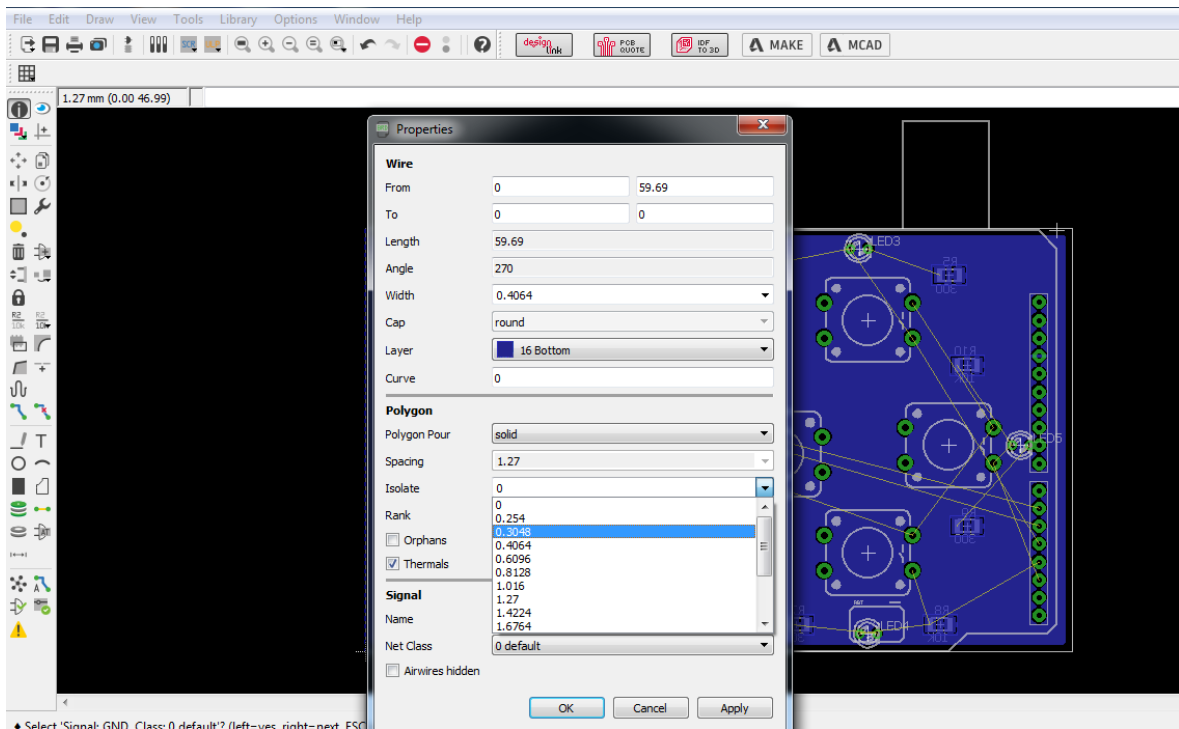
16. Pulso sobre “R2|10K” para nombrar el “Plano de Tierra” y le pongo de nombre “GND”.



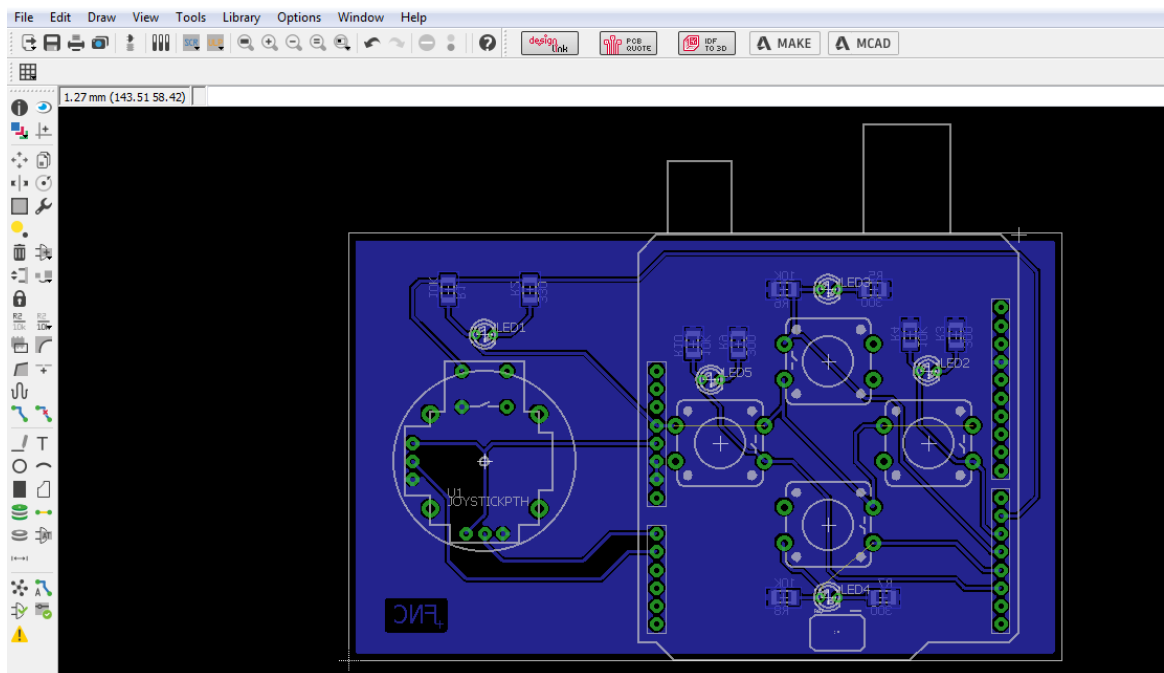
17. Ahora, pulso sobre “Ratsnest” y la parte de GND se me pone de color azul.



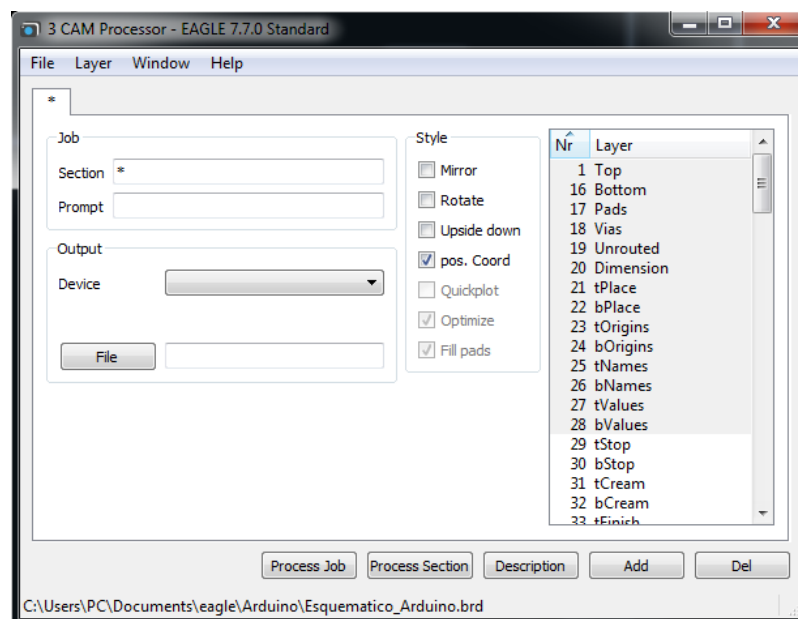
18. Una vez hecho el **"Ratsnest"**, pulso en **"Info"** sobre el recuadro de la PCB y le pongo un **"Isolate"** igual a 0.3048.



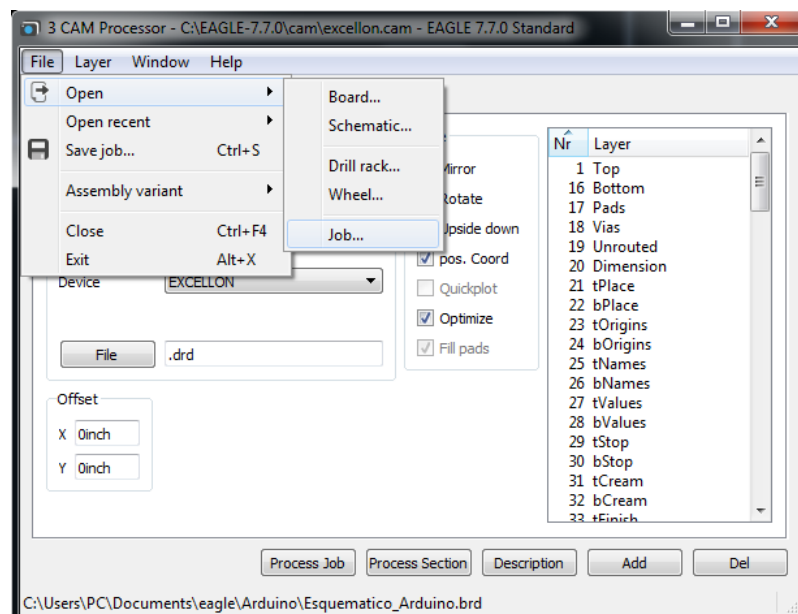
19. Por último, lo que falta es enrutar los caminos y comprobar si hay errores en el diseño. Lo compruebo con el botón **"ERC o Verificación de Errores"**.



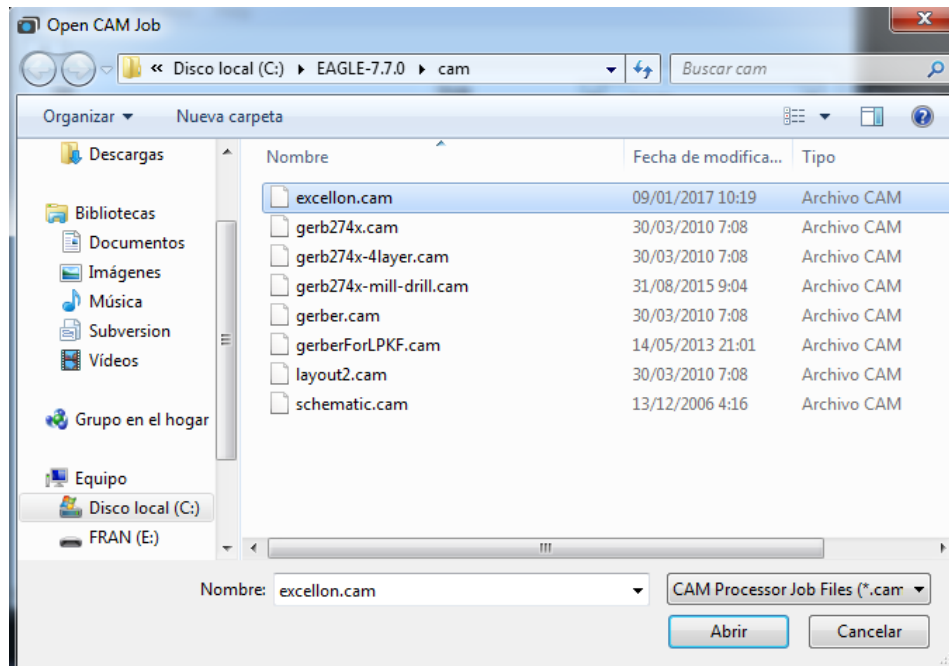
20. Una vez que he finalizado el diseño del “Esquemático” y “Layout” paso a generar los gerbers para poder fabricar la PCB.
21. Primero, me descargo dos librerías necesarias para poder generar los gerbers, que son: “**excellon.cam**” y “**gerberForLPKF.cam**” y los añado en la carpeta “**cam**” de Eagle.
22. Ahora, paso a generarlos. **File -> Cam Processor...** y se abre la siguiente ventana.



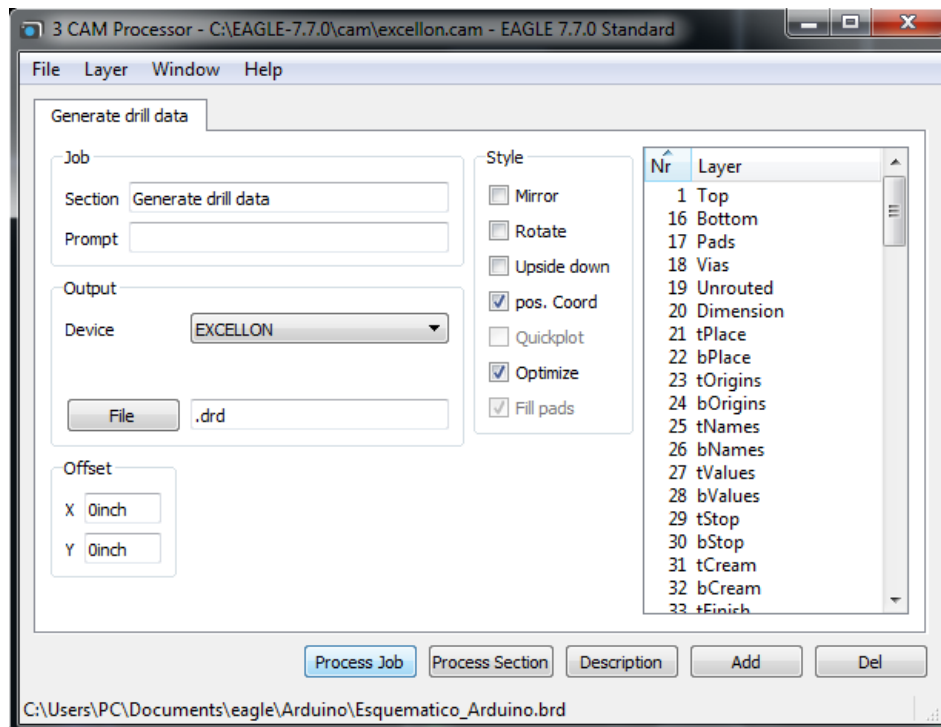
23. Añado la primera librería. **File -> Open -> Job...**



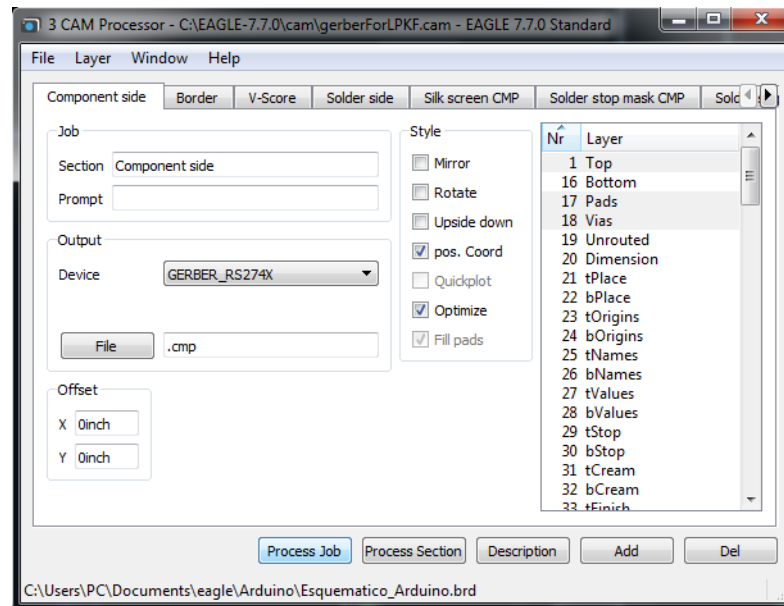
24. Busco en la carpeta “cam” de Eagle la librería “excellon.cam” y la abro.



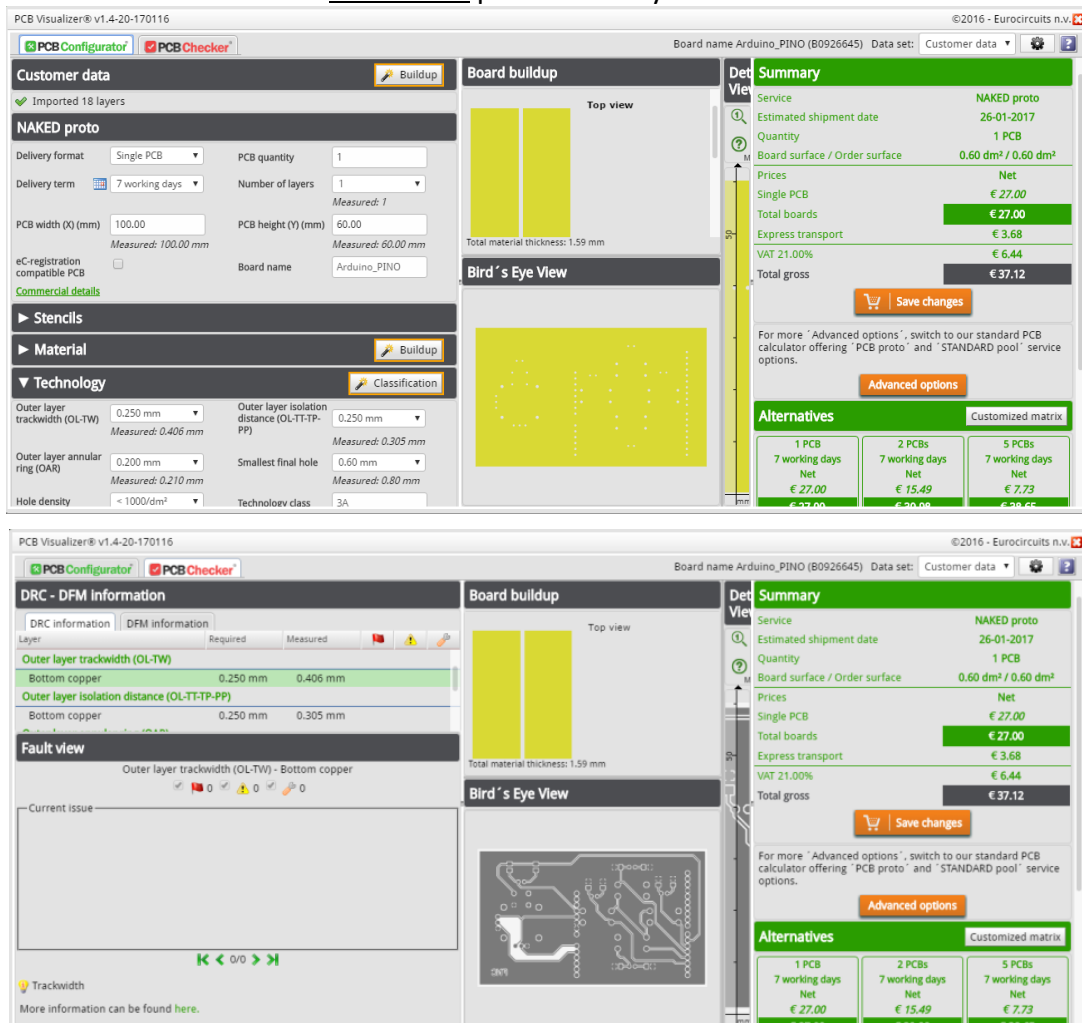
25. Cargo la librería y solo falta pulsar sobre “**Process Job**” para generarlo.



26. Hago lo mismo con la otra librería “gerberForLPCF.cam” para generar todos los gerbers.



27. Ya lo último que falta es coger el archivo generado “Arduino.brd” y mandarlo a [Eurocircuit](http://Eurocircuit.com) para ver si hay errores o no en el diseño.

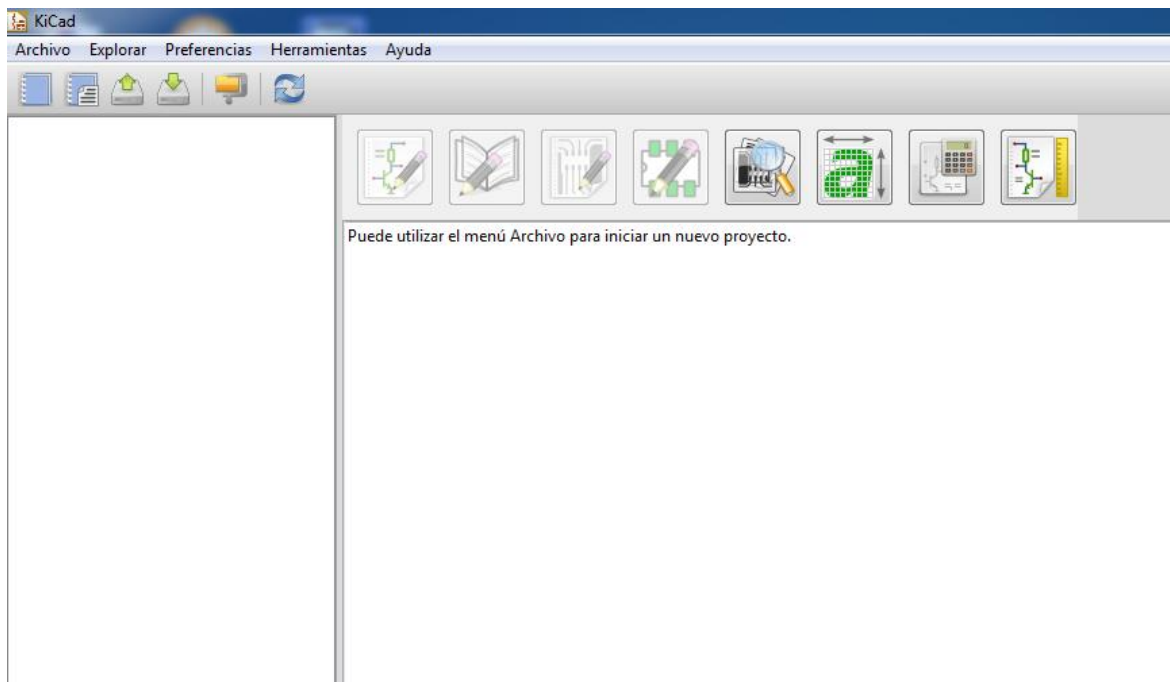


- **¿Qué es KiCad?**

Es un entorno de software usado para el diseño de circuitos electrónicos, muy flexible y adaptable, en el que se pueden crear y editar un gran número de componentes y usarlos en Eeschema. Además, permite el diseño de circuitos impresos modernos de forma sencilla e intuitiva.

- **Utilización del Software**

1. Primero, nos descargamos e instalamos KiCad.
2. Ejecutamos el programa y se nos debería de mostrar la siguiente imagen.



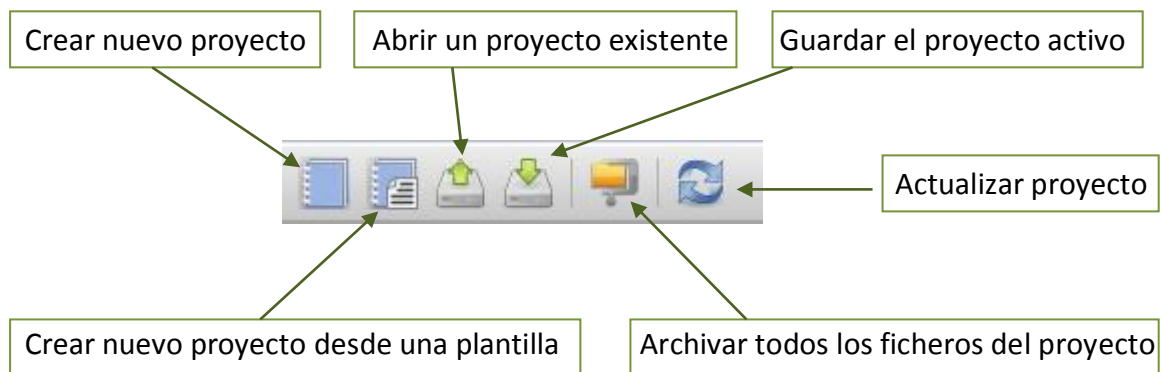
En el panel superior, observamos las siguientes pestañas:

- **Archivo:** Sirve para crear o abrir un proyecto, empaquetar o desempaquetar y cerrar el proyecto.
- **Explorar:** Sirve para abrir un editor de texto o abrir un archivo local.
- **Preferencias:** Sirve para configurar rutas, seleccionar el editor de texto, visor de PDFs y elegir el idioma.

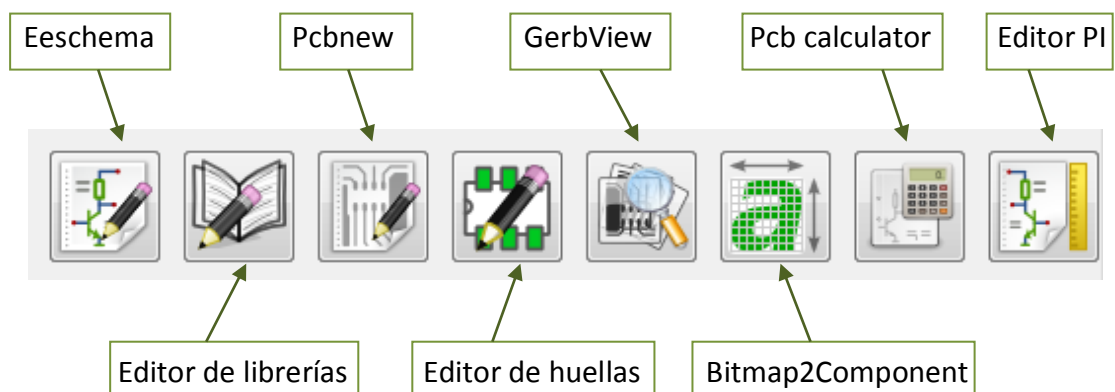
→ **Herramientas:** Sirve para abrir un Eeschema, el editor de librerías, abrir PCBnew, editor de huellas, GerbView, Bitmap2Component, etc.

→ **Ayuda:** Sirve para dar ayuda sobre KiCad en general, manual inicial sobre KiCad y copia de la información sobre la versión.

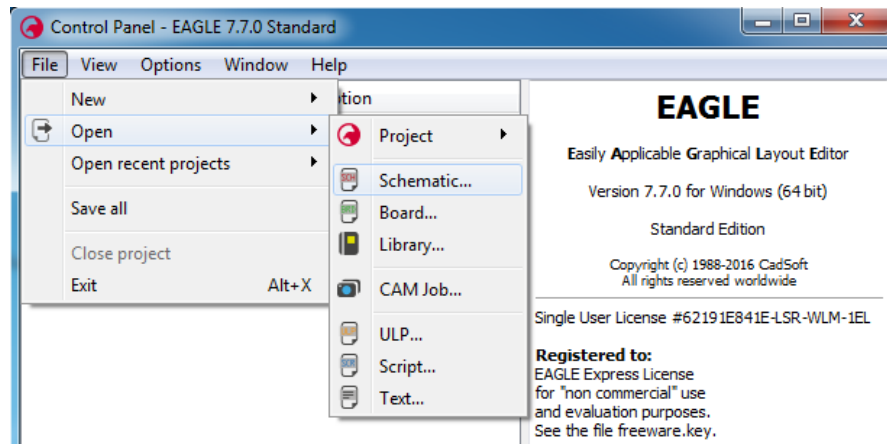
El siguiente panel representa lo siguiente:



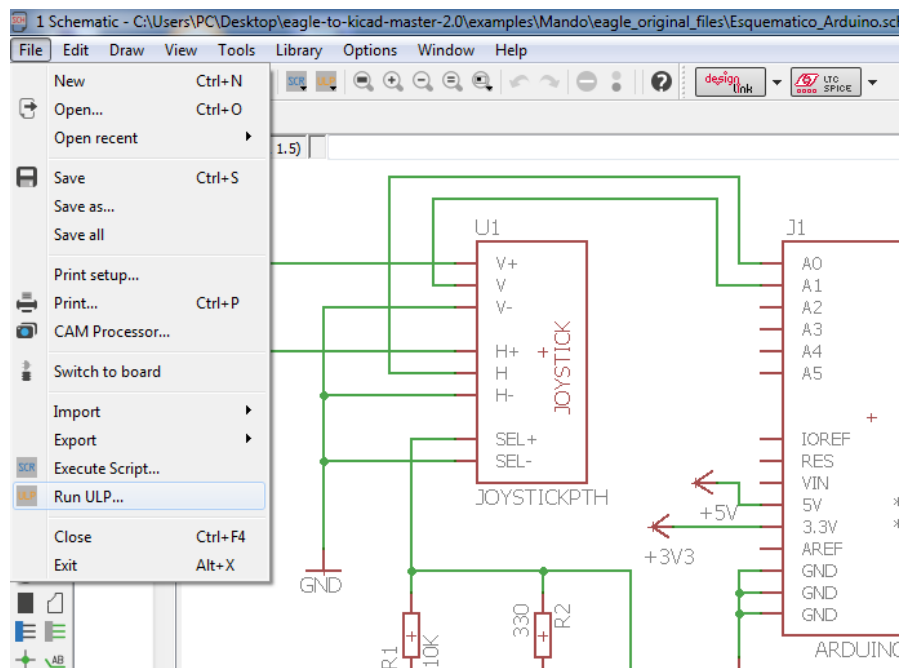
En el panel central, observamos lo siguiente:



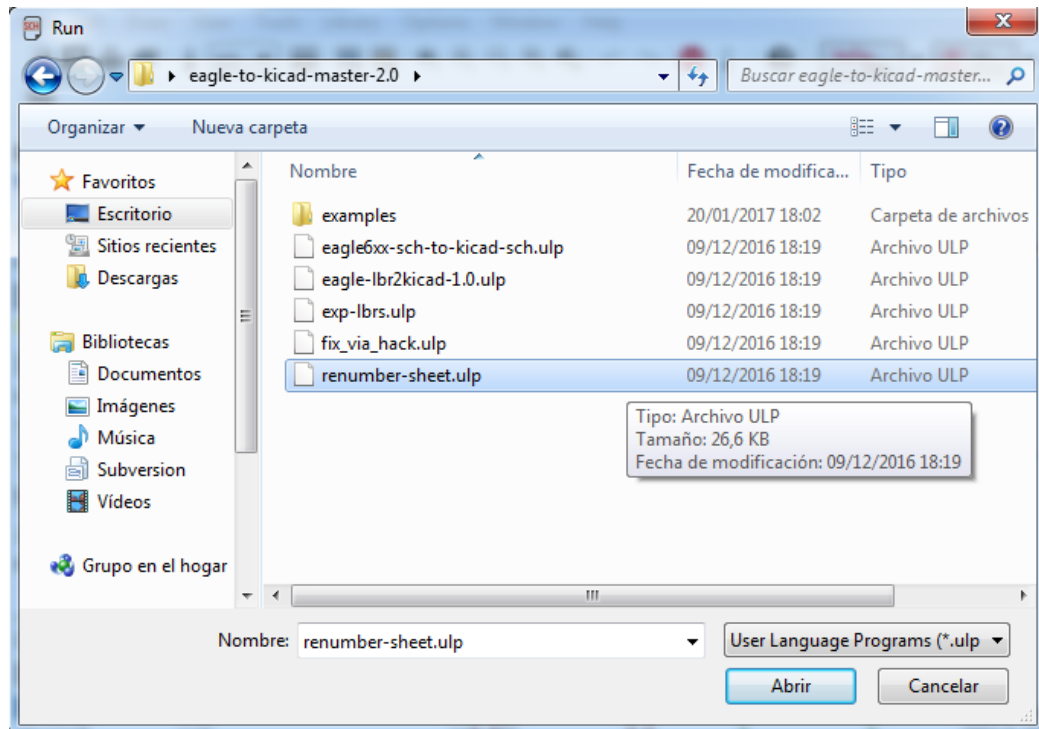
3. Para realizar el diseño del Esquemático y Layout en KiCad he utilizado una técnica que convierte los diseños de Eagle a KiCad.
4. Me descargo una librería que se llama “**eagle-to-kicad-master**”.
5. Abro Eagle. **File -> Open -> Schematic...**



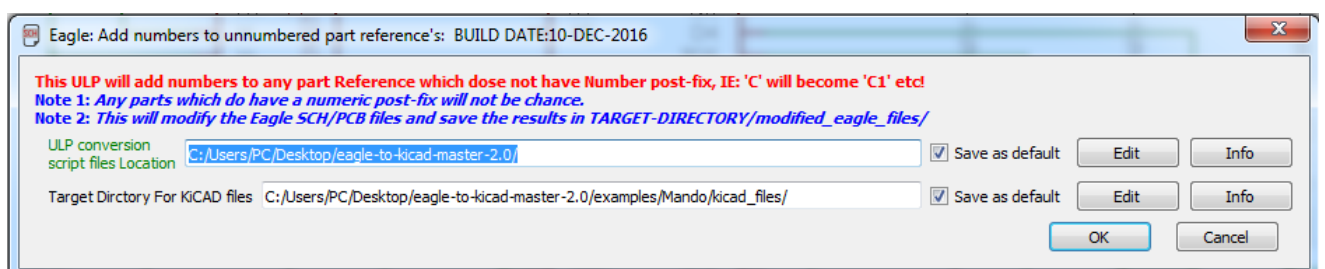
6. Busco el esquemático que he realizado anteriormente y lo abro.
7. Abro el siguiente archivo. **File -> Run ULP...**



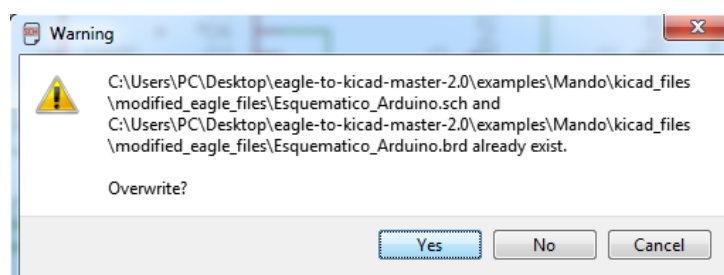
8. Y añadido de la librería que me he descargado anteriormente “renumber-sheet.ulp”.



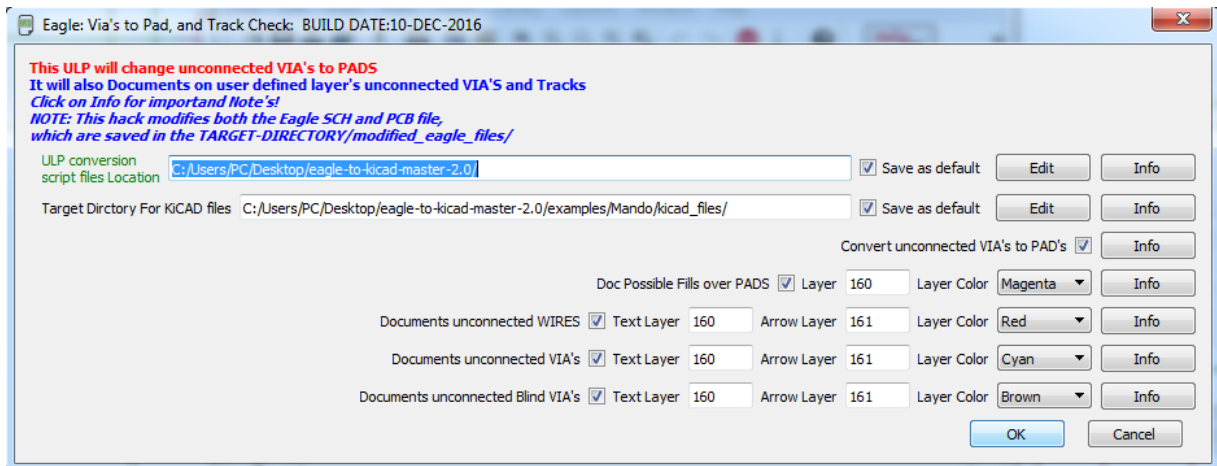
9. Una vez añadido el archivo, se me abre la siguiente ventana. En el que en la primera dirección escribo donde se encuentra el “.ulp” para convertirlo a script. En la segunda dirección escribo el directorio donde quiero que aparezcan los archivos. Pulso sobre “OK”.



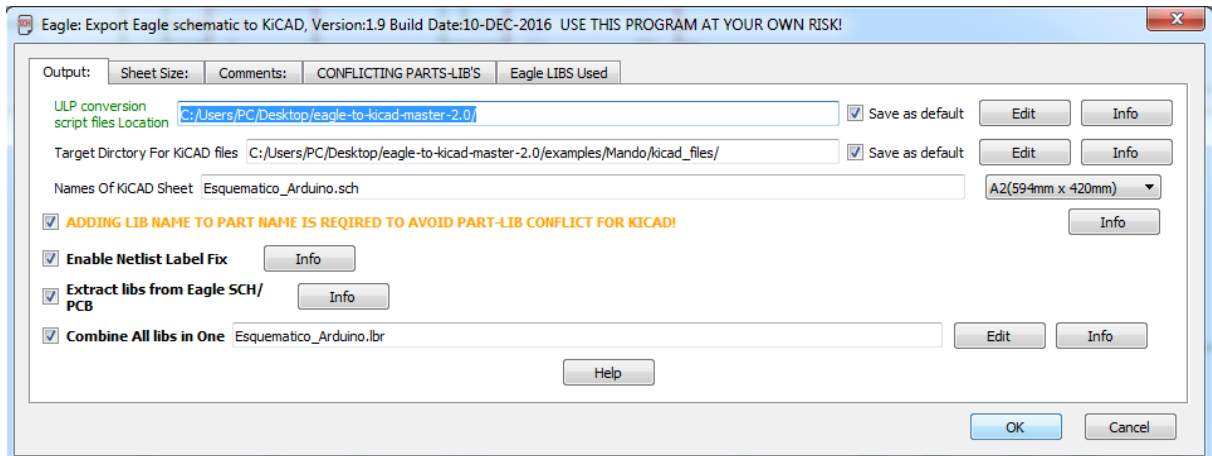
10. Se me va a mostrar el siguiente mensaje de sobre-escritura. Pulso sobre “Yes”.



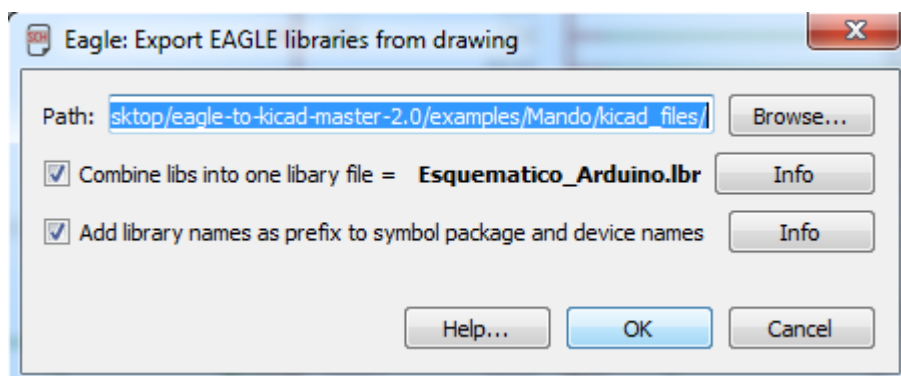
11. A continuación, me sale un nuevo mensaje. Pulsar directamente en **“OK”**. Es para los colores de las vías que van a los pads, etc.



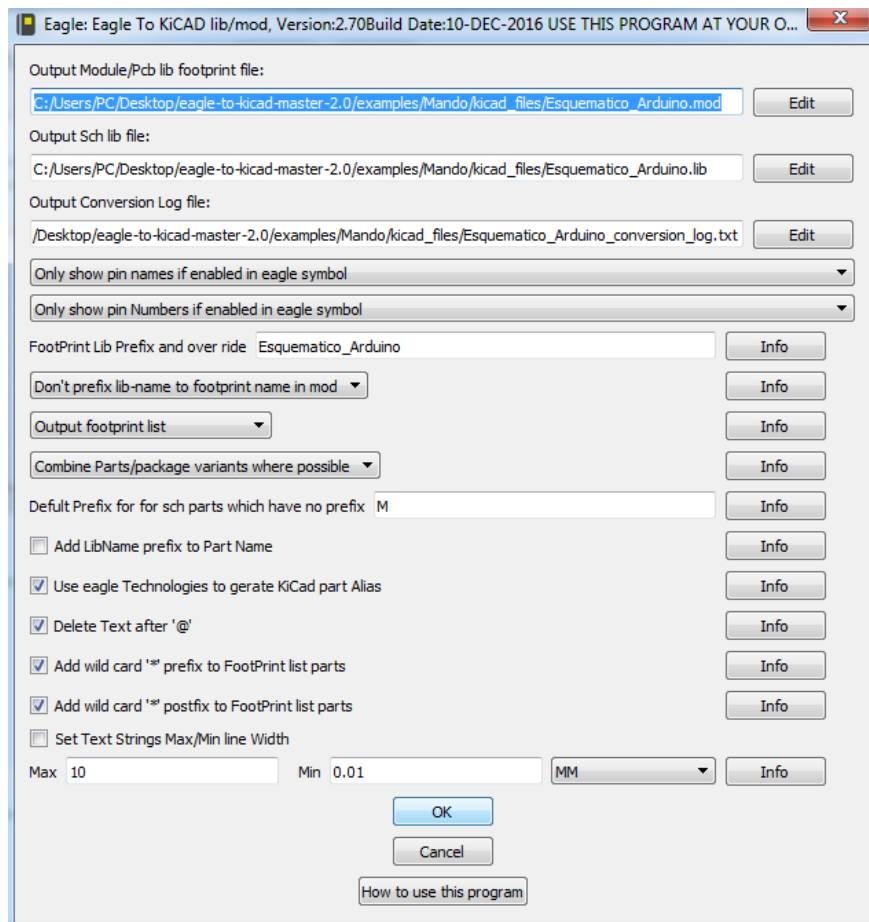
12. Sale un nuevo mensaje y pulsar sobre **“OK”**. Es para exportar el esquemático de Eagle a KiCad.



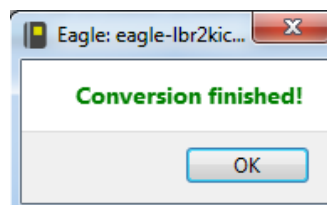
13. Pulsar sobre **“OK”** en el siguiente mensaje. Es para exportar las librerías de Eagle a KiCad.



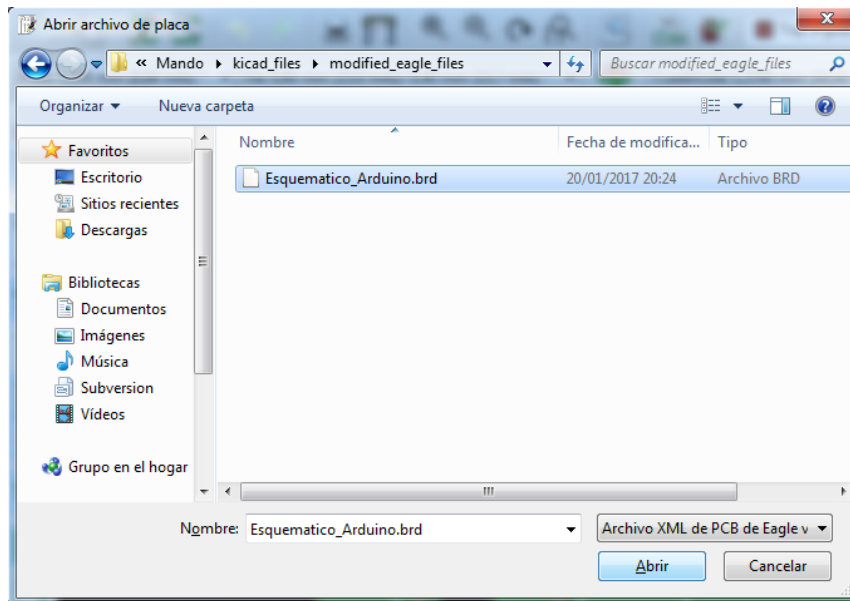
14. Pulsar de nuevo sobre “OK”. Es para exportar los símbolos y módulos de Eagle a KiCad de mi esquemático.



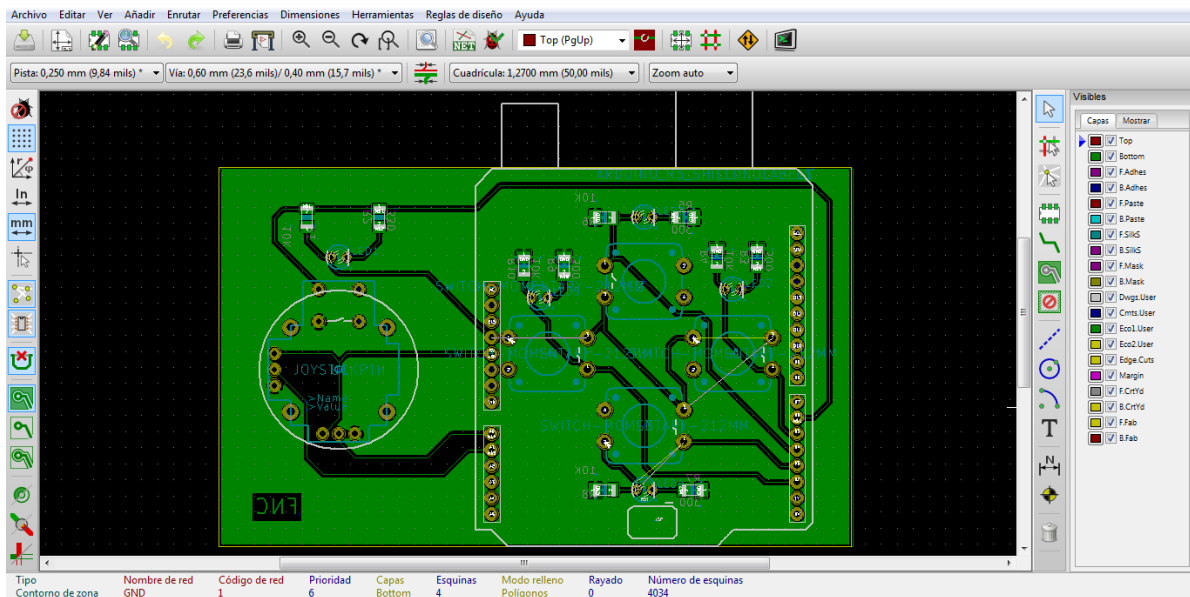
15. Una vez hecho todo el procedimiento me sale el siguiente mensaje. Que la conversión ha terminado.



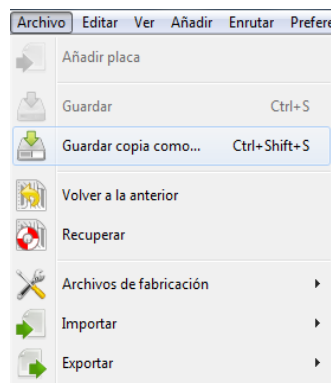
16. En KiCad me creo un “pcbnew” y pulso sobre “Control-L” para buscar el layout (.brd) que tenía creado de Eagle.



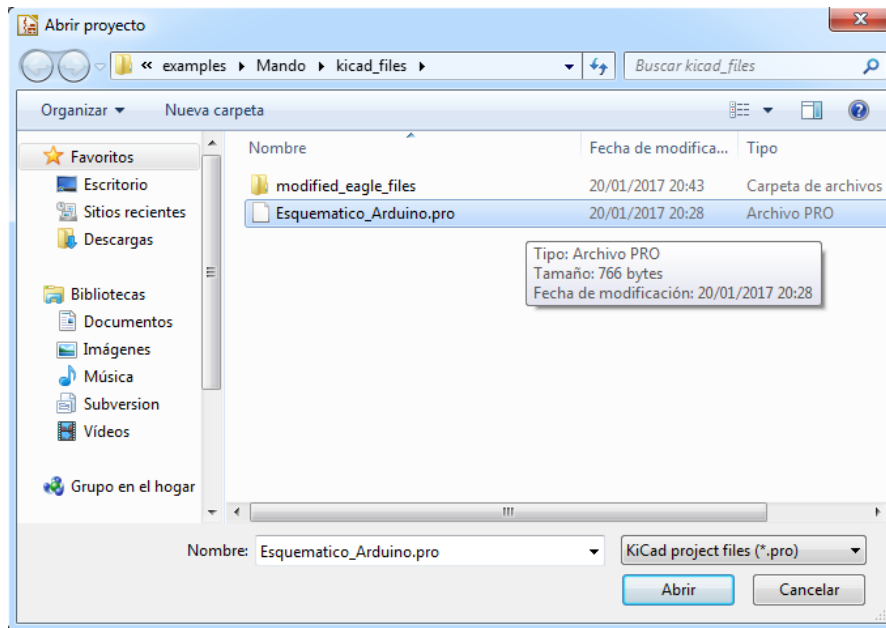
17. Al pulsar sobre “Abrir” se abre el layout en KiCad.



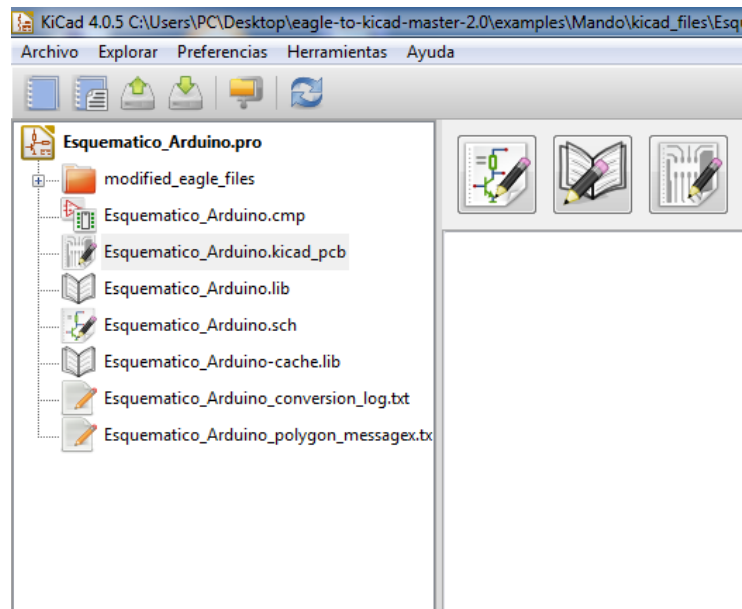
18. Guardamos el archivo “.brd” que es el layout. **Archivo -> Guardar copia como...**



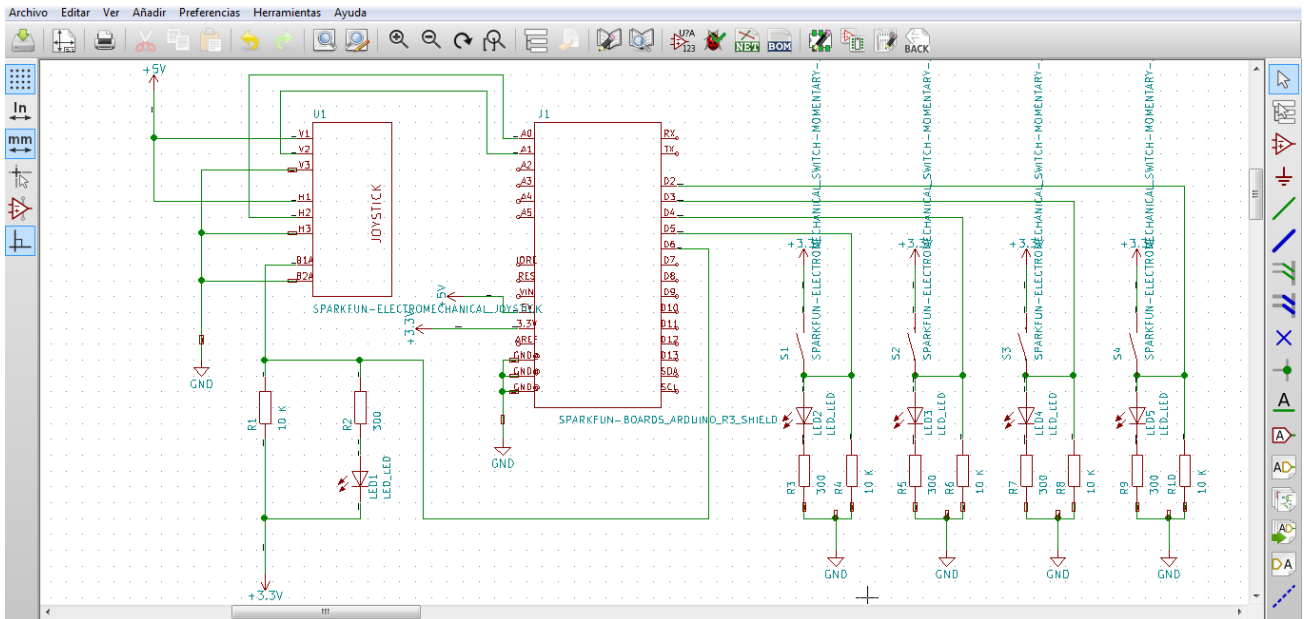
19. Abrimos el proyecto que se nos ha creado. Pulsar sobre “**Abrir**”.



20. En la pantalla principal de KiCad se observa el proyecto que se nos ha creado con todos los archivos necesarios: Esquemático, Layout, librerías importadas, etc.



21. Abro el archivo “Esquematico_Arduino.sch”.



22. Los 2 diseños (Esquemático y Layout) son idénticos a los diseños de Eagle. Lo único que he hecho ha sido importar las librerías de los componentes porque o sino no se verían así.

23. Por último, he comprobado en ambos diseños que no hubiese errores.

○ **Comparativa Eagle y KiCad**

Eagle	
Ventajas	Desventajas
Herramienta cuya relación calidad/precio son muy aceptables.	La simulación se logra por medio de LTspice IV, es decir, con un programa ajeno a Eagle lo que dificulta este proceso.
Interfaz ordenada.	La visualización 3D también se lleva a cabo por medio de paquetes software externos, además de que ni es un proceso automático si no que hay que llevar varios pasos manualmente para poder conseguir una visualización de calidad.
Cuenta con una cantidad enorme de librerías.	Debido a esto, casi siempre resulta muy complicado hacer coincidir las librerías de ambos programas para que se lleve a cabo la tarea, lo que aumenta el tiempo de realización.
Muy utilizado en el mundo, por lo que los usuarios contribuyen con la creación de diagramas y librerías muy frecuentemente.	
La versión free cumple con los requisitos básicos del diseñador, por lo que no es tan necesario conseguir las versiones profesionales.	
Sencillo editor de componentes y librerías.	
Actualizaciones frecuentemente.	

KiCad	
Ventajas	Desventajas
Funciona en las 3 plataformas (Windows, Linux y Mac).	Interfaz muy básica, puede resultar tedioso encontrar alguna función en específico.
Open Source	Transformación Esquemático-Board es compleja.
Permite la creación de diagramas con hasta 16 capas	
Bitmap2Component: crea un componente esquemático o capa a partir de una imagen bmp.	
PcbCalculator: es una herramienta para calcular anchos de pistas dependiendo de la corriente, reguladores de tensión, etc.	

- **Fase de Fabricación**

En esta fase se va a fabricar la PCB y quedará lista para añadirle los componentes.

1. Una vez generado los gerbers, hay que mandarle los siguientes archivos al profesor para que fabrique la PCB.

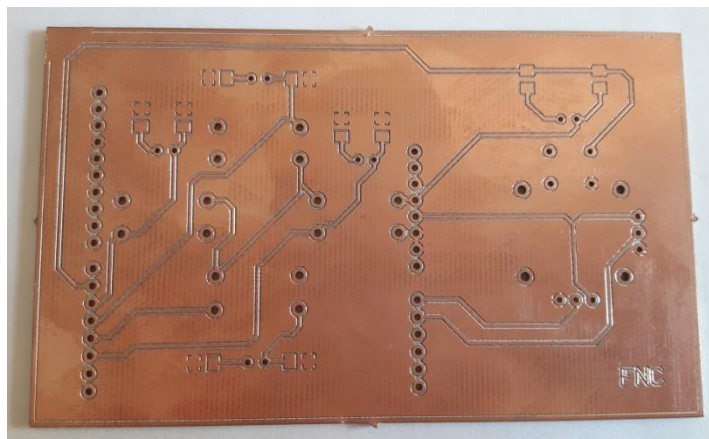
Ficheros gerbers:

- ***.cmp**
- ***.bor**
- ***.dri**
- ***.drd**
- ***.sol**

2. Luego, el profesor utiliza una técnica de traslado del patrón de circuito al sustrato, que es mediante una **“fresadora”**.



3. Este es el resultado de mi PCB fabricada.



- **Fase de Ensamblaje**

En esta fase se va a ensamblar los componentes electrónicos a la PCB.

La PCB que se ha fabricado es de una cara, porque la interconexión de los elementos se hace en una sola cara del sustrato, que es la cara de soldadura (solder side). Los componentes se colocan en la otra cara del sustrato.

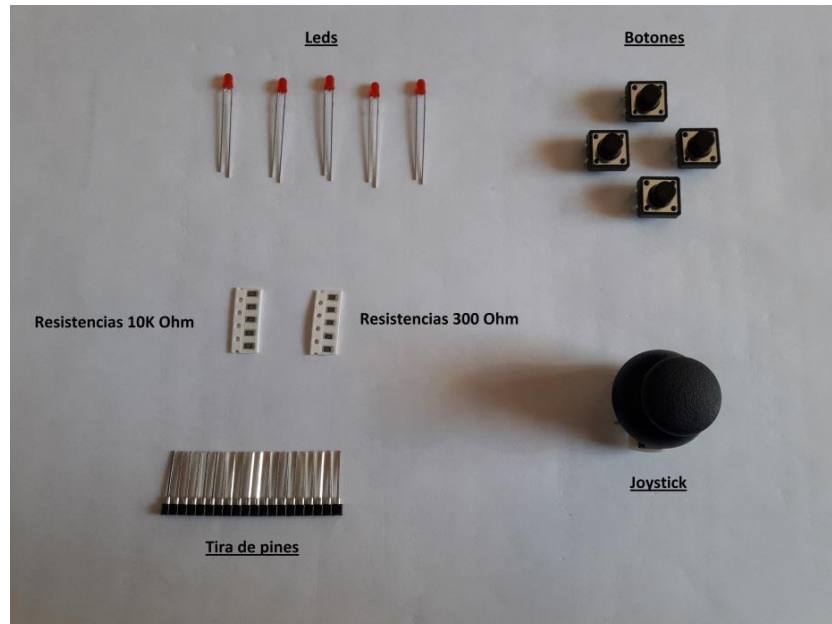
1. Primero, con un polímetro he comprobado si en la PCB hay cortocircuitos en los caminos o en los pads.



2. Una vez, que se ha comprobado que no hay cortocircuitos, paso a soldar los componentes. Los materiales que se han utilizado para soldar son: estación de soldadura (contiene un soldador en forma de lápiz más una consola de control de temperatura) y aleación de soldadura (estaño).



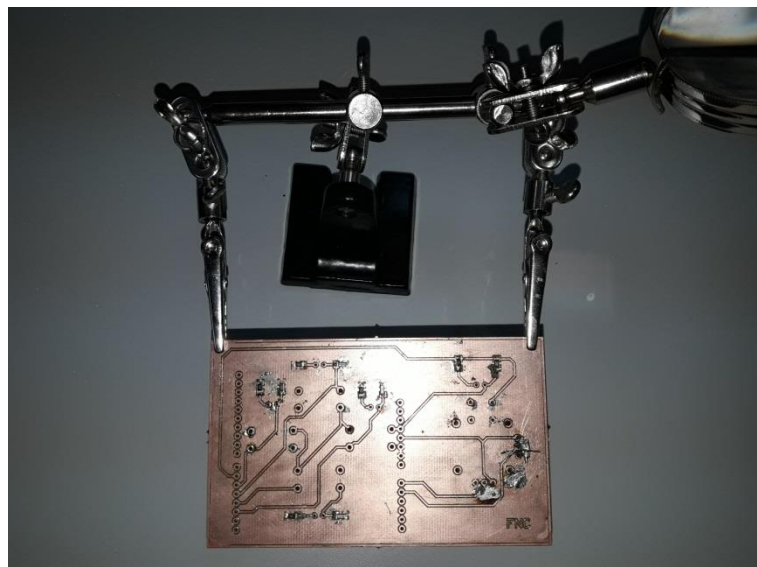
3. Los componentes a soldar son los siguientes: resistencias, leds, botones, joystick y tira de pines.



4. Antes de echar el estaño en la PCB, es aconsejable echar primero en la zona a soldar "Flux" para reducir la tensión superficial del fundente.



5. Es aconsejable comenzar a soldar las resistencias porque al no tener todavía ningún componente soldado la placa va a estar plana y además que son muy pequeñas. Hay que colocar cada resistencia (10K o 300 Ohm) en su lugar correcto para que no haya problemas.

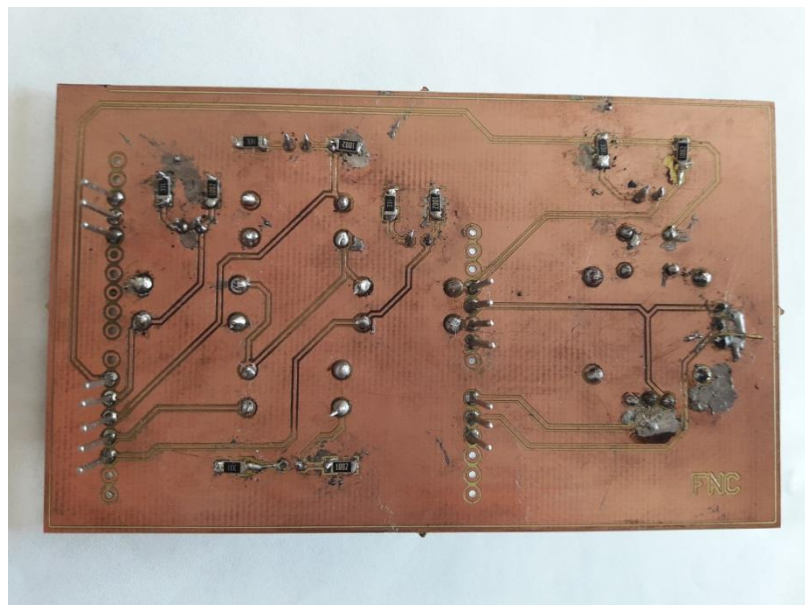


6. Si se da el caso en el que se hecha un pegote grande de estaño y se quiere quitar se puede hacer con una malla para que lo absorba.

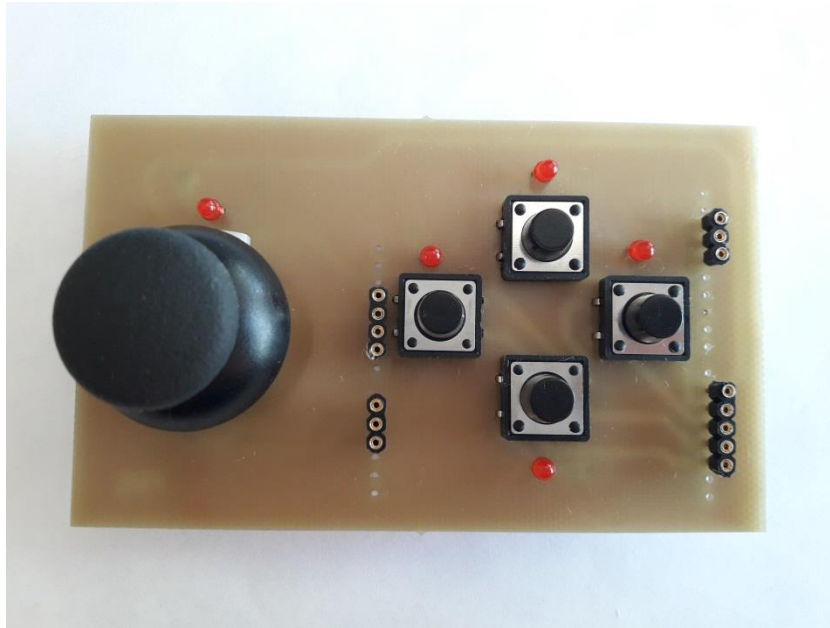


7. Una vez, que he terminado de soldar todos los componentes, el resultado de la placa es el siguiente:

- Cara de soldadura (sustrato)



- Cara de componentes



8. Por último, falta aplicarle un limpiador o disolvente de flux a la placa. Debido a que el flux es activo, puede ir corroyendo el cobre y degenerar las pistas y para evitarlo se le aplica este tipo de limpiador.



Nota: A la hora de realizar la fase de ensamblaje he tenido algunos problemas. Una vez que me dio el profesor la placa fabricada, yo en mi casa por mi cuenta me puse a ensamblarla con un soldador de pistola y con estaño para tuberías (“a lo bestia” y no había soldado en mi vida). Lo ensamblé todo, pero el problema es que no probé con el polímetro si había o no cortocircuitos, y cuando fui a la facultad y probé allí si había cortocircuitos me di cuenta que se me producían muchos. La solución que empleé fue con la malla quitar todo el estaño de la placa y quitar los componentes electrónicos y volverlo a ensamblar todo de nuevo, pero con las herramientas del laboratorio. Me costó muchos menos que cuando lo estaba haciendo en mi casa y así pude salvar la placa y que el profesor no me tuviera que fabricar otra.

- **Fase de Test**

En esta fase se va a realizar la comprobación definitiva de que el SE cumple las especificaciones específicas.

Realmente en esta fase se requieren de varios test:

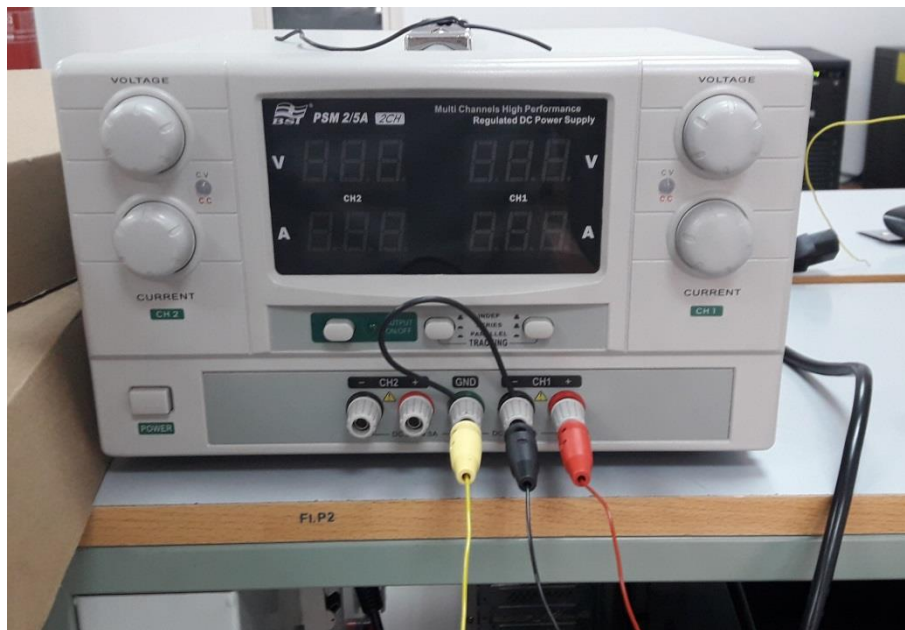
- Verificación de la PCB desnuda (sin componentes)

Se realiza antes del ensamblaje de los componentes en la PCB. Se hace de manera visual y con polímetro en el que se van comprobando si existen cortos, o pistas rotas, en el caso de PCBs de doble cara, comprobar la conexión de las vías. Si es posible se deben arreglar los problemas detectados para que no haya problemas más adelante.

- Verificación de PCB con componentes

Se realiza una vez que se han ensamblado todos los componentes electrónicos a la PCB. Se debe tener preparado un test basado en aplicación de patrones para comprobar la correcta funcionalidad de la placa diseñada.

El profesor me probó con un polímetro que funcionaban todos los botones y el joystick, llegando a 3.3 V. Además, le puso tensión a la PCB con una fuente de alimentación y al pulsar los botones se encendían los leds correspondientes y al pulsar el joystick se encendía también el led correspondiente.



Pruebas con Arduino

Una vez que se ha realizado todo el diseño de la PCB pasando por las diferentes fases (Diseño, Fabricación, Ensamblaje y Test) se va a realizar las diferentes pruebas con Arduino para probarlo con el PC.

He hecho dos tipos de comprobaciones: como mando de juegos (joystick) y como ratón.

- **PCB como mando de juegos**

1. Primero, me descargo la librería “**UnoJoyWin**”, que contiene los siguientes archivos: **TurnIntoAJoystick** (para usar Arduino como Joystick), **TurnIntoAnArduino** (para usar Arduino en modo normal), **UnoJoyDriverInstaller** (para instalar los drivers necesarios), etc.

Nombre	Fecha de modifica...	Tipo	Tamaño
ATmega8u2Code	16/02/2013 19:31	Carpeta de archivos	
Drivers	30/10/2012 4:09	Carpeta de archivos	
Examples	21/10/2012 18:30	Carpeta de archivos	
UnoJoyArduinoSample	21/10/2012 18:33	Carpeta de archivos	
UnoJoyProcessingVisualizer	20/01/2013 9:24	Carpeta de archivos	
README-GettingStarted	28/05/2012 16:41	Documento de tex...	4 KB
TurnIntoAJoystick	03/06/2012 7:47	Archivo por lotes ...	2 KB
TurnIntoAnArduino	28/05/2012 16:41	Archivo por lotes ...	2 KB
UnoJoy	20/01/2013 2:53	Archivo H	11 KB
UnoJoyDriverInstaller	28/05/2012 16:41	Archivo por lotes ...	1 KB

2. También, me descargo los archivos “**JoystickShield_Unojoy.ino**” y “**UnoJoy.h**” para usarlos en Arduino.

El código de **JoystickShield_Unojoy.ino** es:

```
#include "UnoJoy.h"

void setup(){
    setupPins(); // Subrutina que configura los pines del
                //Arduino
    setupUnoJoy();// Inicializa las funciones do UnoJoy
}

void loop(){
    // Siempre está actualizando o dado a ser enviado al
    computador
    dataForController_t controllerData =
    getControllerData();
    setControllerData(controllerData);
}
```



```
void setupPins(void){
    // Configura los pines digitales 2~12
    // como entrada y como pull-up activado
    for (int i = 2; i <= 12; i++){
        pinMode(i, INPUT);
        digitalWrite(i, HIGH);
    }
}

dataForController_t getControllerData(void){

    // Entorno en el que se almacenan los datos de
    //control
    // Sirve para limpiar el buffer donde los datos de
    //control son almacenados
    dataForController_t controllerData =
    getBlankDataForController();

    // Asocia los pines digitales como botones del UnoJoy
    // A "!" invierte a lectura
    controllerData.triangleOn = !digitalRead(4);
    controllerData.circleOn = !digitalRead(3);
    controllerData.squareOn = !digitalRead(6);
    controllerData.crossOn = !digitalRead(5);
    controllerData.selectOn = !digitalRead(2);
    controllerData.startOn = !digitalRead(7);

    // Configura el joystick Analógico
    // La lectura "analogRead(pin)" retorna un valor de
    10 bits (0~1023),
    // Usamos un "map" para abandonar la pista de 8
    // bits (0~255)

    controllerData.leftStickX =
    map((analogRead(A0)),0,1023,255,0);
    controllerData.leftStickY =
    map((analogRead(A1)),0,1023,255,0);

    return controllerData;
}
```

El código de **UnoJoy.h** es:

```
#ifndef UNOJOY_H
#define UNOJOY_H
    #include <stdint.h>
    #include <util/atomic.h>
    #include <Arduino.h>

    typedef struct dataForController_t
    {
        uint8_t triangleOn : 1; // Each of these member
variables
        uint8_t circleOn : 1;    // control if a button
is off or on
        uint8_t squareOn : 1;    // For the buttons,
        uint8_t crossOn : 1;    // 0 is off
        uint8_t l1On : 1;       // 1 is on
        uint8_t l2On : 1;
        uint8_t l3On : 1;       // The : 1 here just
tells the compiler
        uint8_t r1On : 1;       // to only have 1 bit
for each variable.
                                // This saves a lot of
space for our type!
        uint8_t r2On : 1;
        uint8_t r3On : 1;
        uint8_t selectOn : 1;
        uint8_t startOn : 1;
        uint8_t homeOn : 1;
        uint8_t dpadLeftOn : 1;
        uint8_t dpadUpOn : 1;
        uint8_t dpadRightOn : 1;

        uint8_t dpadDownOn : 1;
        uint8_t padding : 7;    // We end with 7 bytes
of padding to make sure we get our data aligned in bytes

        uint8_t leftStickX : 8; // Each of the analog
stick values can range from 0 to 255
        uint8_t leftStickY : 8; // 0 is fully left or
up
        uint8_t rightStickX : 8; // 255 is fully right
or down
        uint8_t rightStickY : 8; // 128 is centered.
                                // Important -
analogRead(pin) returns a 10 bit value, so if you're
getting strange
                                // results from
analogRead, you may need to do (analogRead(pin) >> 2) to
get good data
    } dataForController_t;
```

```
    // Call setupUnoJoy in the setup block of your
program.
    // It sets up the hardware UnoJoy needs to work
properly
    void setupUnoJoy(void);

    // You can also call the set
void setupUnoJoy(int);

    // This sets the controller to reflect the button
and
    // joystick positions you input (as a
dataForController_t).
    // The controller will just send a zeroed (joysticks
centered)
    // signal until you tell it otherwise with this
function.
    void setControllerData(dataForController_t);

    // It returns a dataForController_t, so you want to
call it like:
    // myControllerData =
getBlankDataForController();
    dataForController_t getBlankDataForController(void);

    dataForController_t controllerDataBuffer;

    void setControllerData(dataForController_t
controllerData){

        ATOMIC_BLOCK(ATOMIC_FORCEON){
            controllerDataBuffer = controllerData;
        }
    }

    volatile int serialCheckInterval = 1;
    // This is an internal counter variable to count ms
between
    // serial check times
    int serialCheckCounter = 0;

    void setupUnoJoy(void){
        // First, let's zero out our controller data buffer
(center the sticks)
        controllerDataBuffer = getBlankDataForController();

        Serial.begin(38400);

        OCR0A = 128;
        TIMSK0 |= (1 << OCIE0A);
    }
```

```
void setupUnoJoy(int interval){
    serialCheckInterval = interval;
    setupUnoJoy();
}

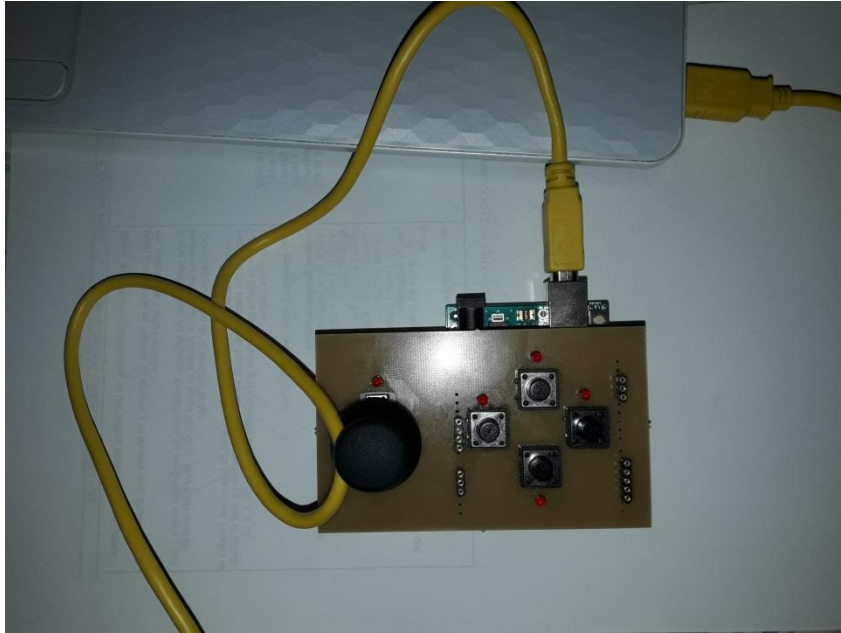
ISR(TIMER0_COMPA_vect){
    serialCheckCounter++;
    if (serialCheckCounter >= serialCheckInterval){
        serialCheckCounter = 0;
        while (Serial.available() > 0) {
            pinMode(13, OUTPUT);
            //digitalWrite(13, HIGH);
            // Get incoming byte from the ATmega8u2
            byte inByte = Serial.read();
            // That number tells us which byte of the
dataForController_t struct
            // to send out.

Serial.write(((uint8_t*)&controllerDataBuffer)[inByte]);
            //digitalWrite(13, LOW);
        }
    }
}

dataForController_t getBlankDataForController(void){
    // Create a dataForController_t
    dataForController_t controllerData;
    // Make the buttons zero
    controllerData.triangleOn = 0;
    controllerData.circleOn = 0;
    controllerData.squareOn = 0;
    controllerData.crossOn = 0;
    controllerData.l1On = 0;
    controllerData.l2On = 0;
    controllerData.l3On = 0;
    controllerData.r1On = 0;
    controllerData.r2On = 0;
    controllerData.r3On = 0;
    controllerData.dpadLeftOn = 0;
    controllerData.dpadUpOn = 0;
    controllerData.dpadRightOn = 0;
    controllerData.dpadDownOn = 0;
    controllerData.selectOn = 0;
    controllerData.startOn = 0;
    controllerData.homeOn = 0;
    //Set the sticks to 128 - centered
    controllerData.leftStickX = 128;
    controllerData.leftStickY = 128;
    controllerData.rightStickX = 128;
    controllerData.rightStickY = 128;
    // And return the data!
    return controllerData;
}

#endif
```

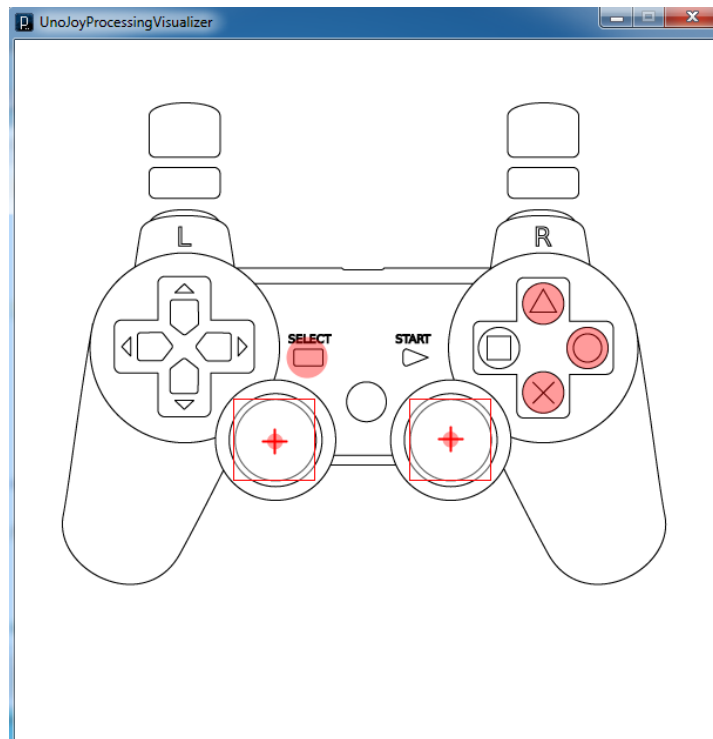
3. Hay que tener la PCB conectada a Arduino, y ésta al PC.



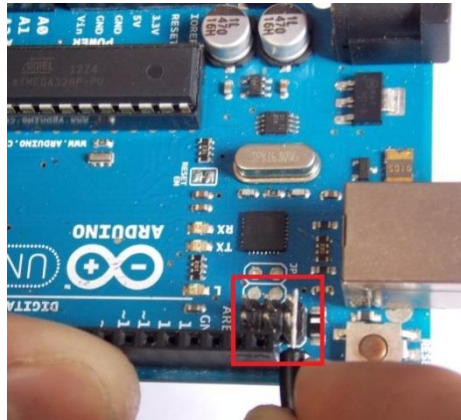
4. Se verifican los dos códigos y se cargan en Arduino.



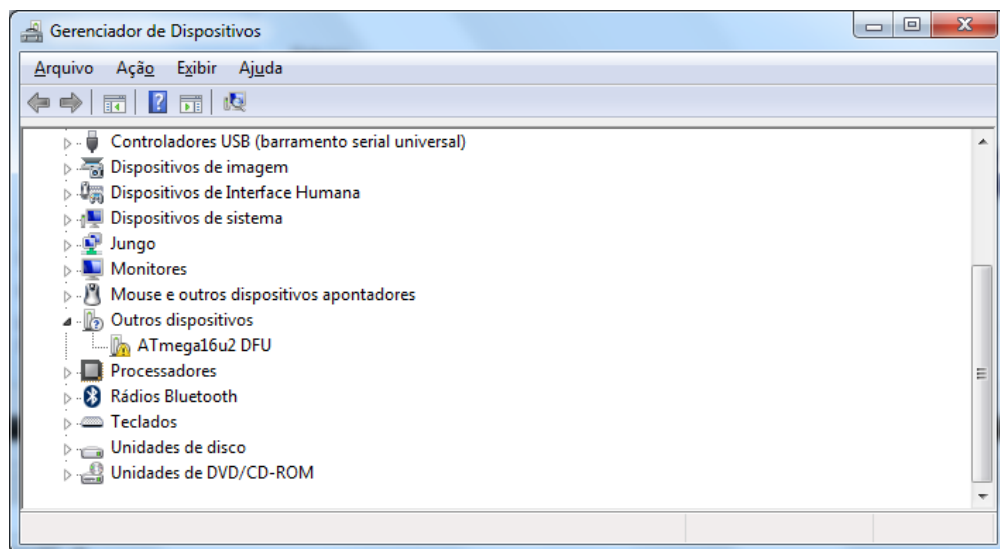
5. Una vez cargado el programa en Arduino paso a visualizar el mando. Entro en la carpeta **“UnoJoyWin”** -> **“UnoJoyProcessingVisualizer”** -> **“UnoJoyProcessingVisualizer.exe”**. Se observa lo siguiente:



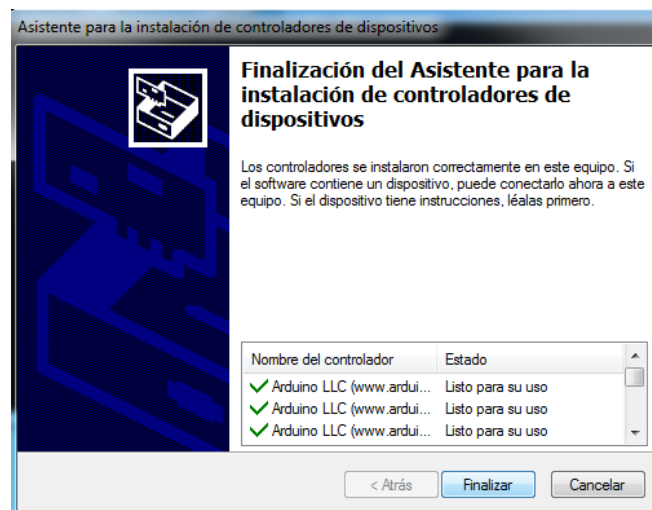
6. Ahora, paso a convertir el Arduino en Joystick. Hago un cortocircuito para reprogramar el chip USB/ Serie Comunicación (ATmega16u2) y ponerlo en modo DFU (Dispositivo de Actualización de Firmware).



7. En el administrador de dispositivos se reconocerá el Arduino como en la imagen:



8. Ahora paso a instalar los drivers. Ejecuto “UnoJoyDriverInstaller.bat”



9. Ejecuto “**TurnIntoAJoystick.bat**” para convertir el Arduino como joystick.

```

C:\ ATMEGA16U2 Command Line Interpreter
-hardware usb -operation erase f memory flash blankcheck loadbuffer "UnoJoy.hex"
" program verify start reset 1024
Running batchisp 1.2.5 on Mon Jan 23 21:17:34 2017

ATMEGA16U2 - USB - USB/DFU

Device selection..... PASS
Hardware selection..... PASS
Opening port..... PASS
Reading Bootloader version..... PASS      1.2.0
Erasing..... PASS
Selecting FLASH..... PASS
Blank checking..... PASS      0x000000 0x02fff
Parsing HEX file..... PASS      UnoJoy.hex
Programming memory..... PASS      0x000000 0x00ad9
Verifying memory..... PASS      0x000000 0x00ad9
Starting Application..... PASS      RESET 1024

Summary: Total 11 Passed 11 Failed 0
Now, you need to unplug the Arduino and plug it back in,
but it will show up as a joystick! Press any key to exit....

```

10. Desconectar el Arduino del PC y volverlo a conectar.

11. Hacer click en “**Dispositivos e impresoras**” y se observa el Arduino como mando.

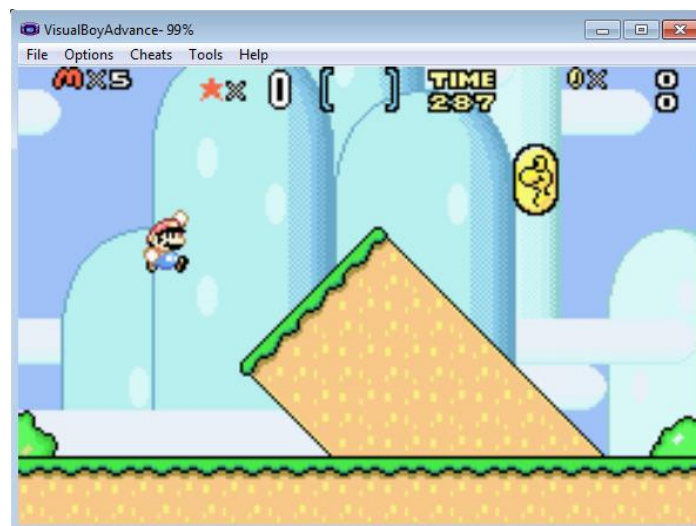


12. Me he descargado un emulador “**WinDS PRO 2017**” para poder probar un juego.

13. Abro el juego y configuro los botones (pulsando cada uno en su lugar).



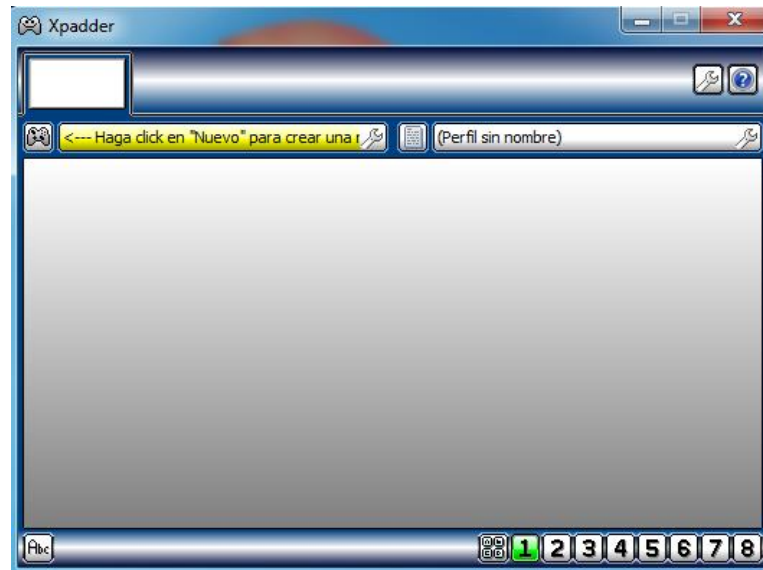
14. Por último, paso a probarlo y verificar que funciona el Arduino con la PCB como mando.



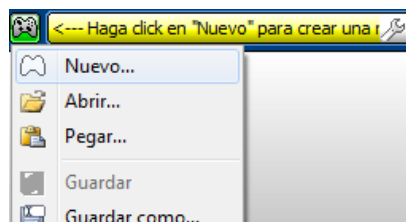
15. Para convertir Arduino en modo normal, hay que hacer de nuevo el paso 6 (hacer un cortocircuito y poner en modo DFU) y luego ejecutar **"TurnIntoAnArduino"**. Hay que desconectar y conectar de nuevo el cable USB del Arduino para que haga efecto los cambios.

- **PCB como ratón**

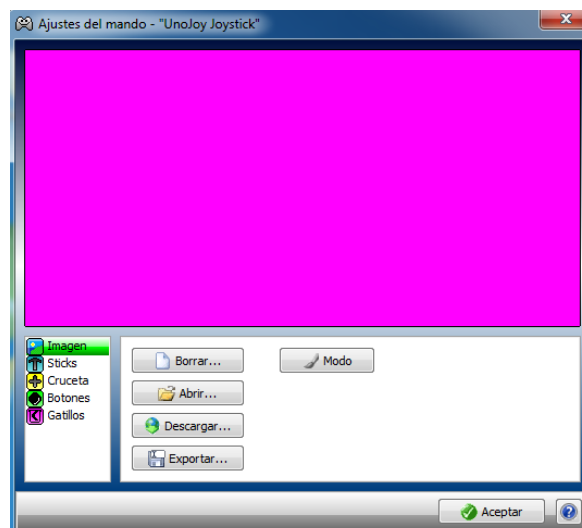
1. Me descargo un programa, llamado **Xpadder**, que sirve para configurar un dispositivo como mando de la play, ratón, teclado, etc.
2. Hago los pasos 1 – 10 del apartado anterior (PCB como mando de juegos) para dejar Arduino como un mando/joystick.
3. Abro **Xpadder** y se observa la siguiente imagen.



4. Hago click en el cuadro amarillo o en el dibujo del mando y “Nuevo” para crearme una configuración nueva.



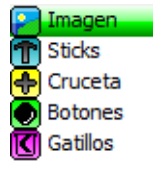
Se me abre la siguiente ventana:



5. Se observan dos paneles.

En el primer panel, se encuentran las siguientes pestañas:

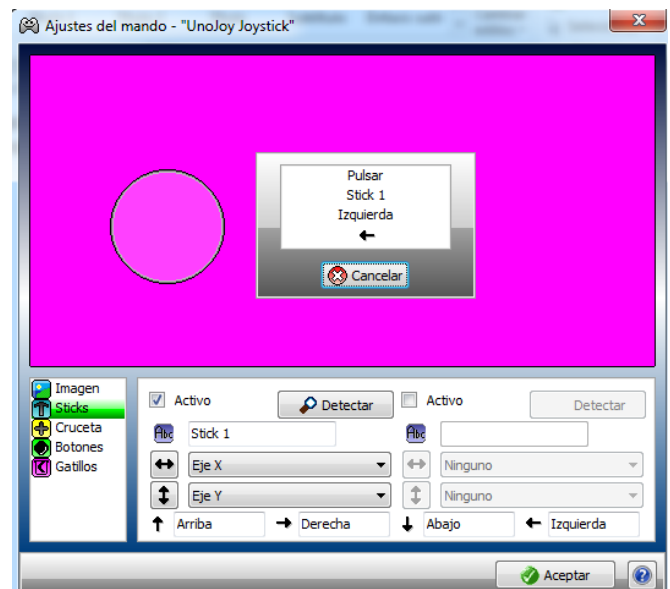
- Imagen: Para cargar una imagen.
- Sticks: Para configurar los movimientos del joystick.
- Cruceta: Para configurar los movimientos de las crucetas.
- Botones: Para configurar la funcionalidad de los botones.
- Gatillos: Para configurar la funcionalidad de los gatillos.



En el segundo panel, dependiendo de la pestaña que se haya elegido saldrán unas configuraciones u otras.

6. Como lo que quiero hacer es que la PCB/mando haga de ratón voy a utilizar el joystick para moverse y dos botones para que haga de botón derecho y de botón izquierdo del ratón.

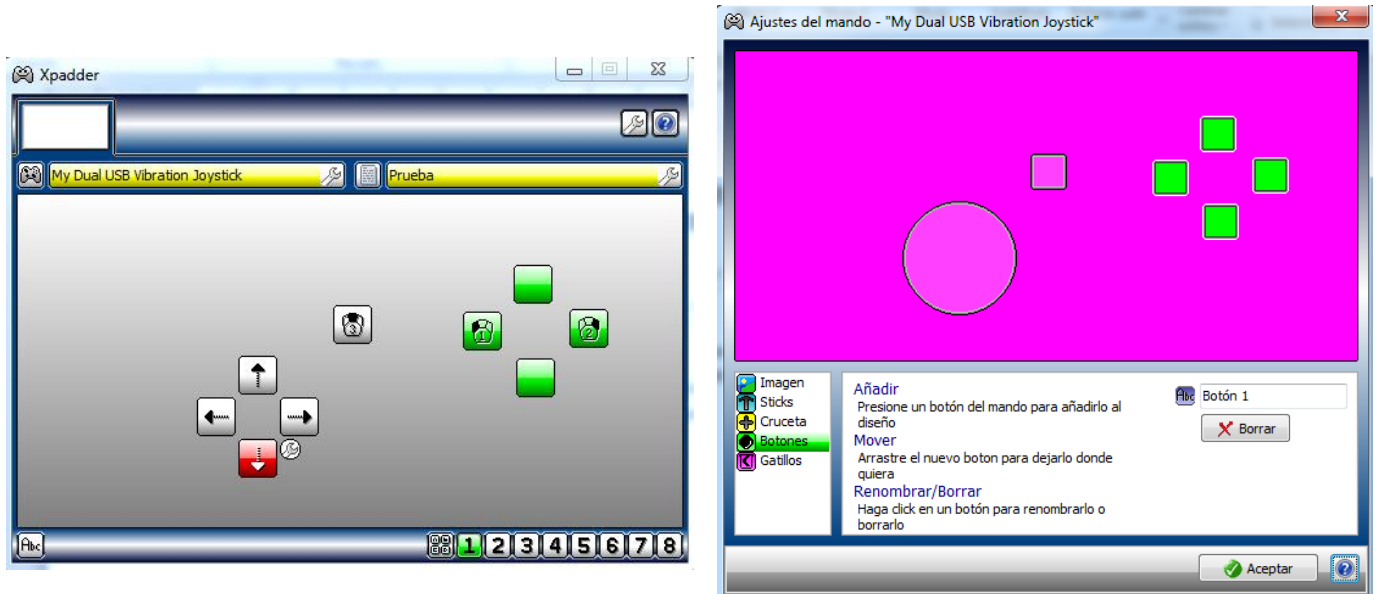
7. Pulso sobre “**Sticks**” y activo el joystick.



8. Dependiendo del mensaje que me salga en la ventana segunda, le doy hacia un lado u otro al joystick para que se me configure.

9. Una vez que haya configurado el joystick para el movimiento del ratón, paso a configurar los botones derecho e izquierdo del ratón.

10. Pulso sobre “**Botones**” y los activos.



11. Por último, guardo la configuración y debería de funcionar. Aquí en una captura es imposible mostrar el movimiento del ratón, para ello se lo he mostrado al profesor en persona.

Conclusión

Con las sesiones de laboratorio de este segundo bloque de la asignatura, he aprendido a utilizar programas de Diseños de PCB (Eagle y KiCad), a ensamblar los componentes electrónicos a la PCB fabricada (nunca había soldado antes) y a probar la placa como si fuera un ratón o un mando de juegos.

Lo que me ha parecido más fácil ha sido el diseño del esquemático a realizar de mi PCB, la verificación de los componentes y la utilización de la placa como mando, porque lo he estado siguiendo de un tutorial de internet.

Lo que me ha parecido más difícil ha sido el diseño de la board (layout) a la hora de enrutar los caminos porque se me cruzaban y soldar los componentes (la primera vez fue un desastre y tuve que empezar todo de nuevo).

Me hubiera gustado ver hecho la placa de expansión también compatible con Papilio, pero por falta de tiempo y con la cantidad de exámenes me ha resultado imposible.

En mi opinión, con 4 años que llevo en la carrera de Ingeniería Informática por la rama de Computadores, las prácticas de esta asignatura (Laboratorio de Desarrollo Hardware) y sobre todo las del segundo bloque, la realización de una placa de expansión para Arduino para poder utilizarla como un mando, me han parecido las más interesantes, entretenidas y productivas de estos 4 años que llevo en la carrera. Ojalá haya más asignaturas de esta forma, porque se aprende mucho más y son más productivas.

Glosario

- **Cortocircuito:** fallo en un aparato o línea eléctrica por el cual la corriente eléctrica pasa directamente del conductor activo o fase al neutro o tierra en sistemas monofásicos de corriente alterna, o entre polos opuestos en el caso de la corriente continua.
- **Eeschema:** es un potente software de captura de esquemas distribuido como parte de KiCad y disponible para Linux, Apple OS X y Windows. Aplicación integrada donde todas las funciones para el dibujado, control, planos, gestión de bibliotecas y acceso al software de diseño del PCB son llevadas a cabo dentro del propio Eeschema.
- **Emulador:** es un software que permite ejecutar programas o videojuegos en una plataforma distinta de aquella para la cual fueron escritos originalmente.
- **Esquemático:** es el diagrama del circuito electrónico donde aparecen los diferentes componentes electrónicos y la interconexión entre ellos.
- **Eurocircuit:** son fabricantes especializados de prototipos y pequeños lotes de PCB.
- **Flux:** sustancia que se emplea en el proceso de soldadura con la misión de eliminar las impurezas y capas de óxido en la superficie del metal, evitar que la base del metal se vuelva a oxidar durante el proceso de soldadura y ayuda en la transferencia de calor al metal que se suelda. La composición es: disolventes y sólidos.
- **Freeware:** es un término que se utiliza para un tipo de software que se distribuye sin costo, disponible para su uso, pero que mantiene el copyright, por lo que no se puede modificar o utilizar libremente como ocurre con el software libre.
- **Fresadora:** es una máquina tipo plotter que hace un “dibujo” sobre la placa empleando fresas que eliminan el cobre de la misma.
- **Gerbers:** es un formato de archivo que contiene la información necesaria para la fabricación de la placa de circuito impreso o PCB.
- **GND (toma de tierra):** se emplea en las instalaciones eléctricas para llevar a tierra cualquier derivación indebida de la corriente eléctrica a los elementos que puedan estar en contacto, ya sea directa o indirectamente, con los usuarios de aparatos de uso normal, por un fallo del aislamiento de los conductores activos, evitando el paso de corriente al posible usuario.

- **Joystick:** es un dispositivo de control de dos o tres ejes. Se suele diferenciar entre digitales (que leen cuatro interruptores encendido/apagado en cruceta situada en la base más sus combinaciones y los botones de acción) y los analógicos (que usan potenciómetros para leer continuamente el estado de cada eje, y además de botones de acción pueden incorporar controles deslizantes).
- **Layout:** es el dibujo donde aparecen los componentes electrónicos con su huella a tamaño real ("footprint") en la posición que van a ocupar en la PCB final y los caminos de interconexión entre pines.
- **Led:** es un diodo que emite luz en polarización directa.
- **Pad:** zona de contacto en la PCB de un pin o terminal de un componente. El pad de Through-hole es circular y el pad de Smd es rectangular.
- **PCB:** es la superficie constituida por caminos, pistas o buses de material conductor laminadas sobre una base no conductora. El circuito impreso se utiliza para conectar eléctricamente a través de las pistas conductoras, y sostener mecánicamente, por medio de la base, un conjunto de componentes electrónicos.
- **Polímetro:** es un instrumento eléctrico portátil para medir directamente magnitudes eléctricas activas, como corrientes y potenciales (tensiones), o pasivas, como resistencias, capacidades y otras.
- **Pulsador:** es un dispositivo utilizado para realizar cierta función. Los botones son por lo general activados, al ser pulsados con un dedo. Permiten el flujo de corriente mientras son accionados. Cuando ya no se presiona sobre él vuelve a su posición de reposo.
- **Resistencia:** oposición que tienen los electrones al moverse a través de un conductor. Responde a la ley de Ohm: $V = I \times R$. Pueden ser fijas o variables.
- **Soldador:** es una herramienta eléctrica usada para soldar. Funciona convirtiendo la energía eléctrica en calor, que a su vez provoca la fusión del material utilizado en la soldadura, como por ejemplo el estaño.
- **Soldadura:** es un proceso para la unión de piezas metálicas mediante el uso de cualquiera de las diversas aleaciones fusibles (estaño (63%) y plomo (37%)), cuya temperatura de fusión es más bajo que el del material a unir, y en el que la superficie de las partes crea un enlace intermolecular, sin llegar a ser derretido. Hay soldadura blanda (por debajo de 450°C) y soldadura dura (por encima de 450°C).

Bibliografía

Objetivos y teoría de PCB

- <https://www.dte.us.es/docencia/etsii/gii-ic/laboratorio-de-desarrollo-hardware/temas/Tema3DPCB/view>
- <https://www.dte.us.es/docencia/etsii/gii-ic/laboratorio-de-desarrollo-hardware/temas/CPCBs/view>
- <https://www.dte.us.es/docencia/etsii/gii-ic/laboratorio-de-desarrollo-hardware/temas/Tema5NormasPCB/view>
- <https://www.dte.us.es/docencia/etsii/gii-ic/laboratorio-de-desarrollo-hardware/lab/PracticaDisPCB/view>

Descarga y tutorial de Eagle

- <https://cadsoft.io/#>
- <https://www.dte.us.es/docencia/etsii/gii-ic/laboratorio-de-desarrollo-hardware/temas/TutEagle/view>

Verificación de la PCB con Eurocircuit

- <https://be.eurocircuits.com/shop/eclogin.aspx>

Librerías Sparkfun Eagle

- <https://mega.nz/#!3s1SFIDI!oYEQK5YwUEjf9LPsmgRBMwtVXGRalHt8OeUAu55xE50>
- <https://mega.nz/#!f88nxLpa!FnshsyYb08qffpBs0PtGtLS37Bkdt6-K3HhoBzLSvbK>

Librerías para generar los Gerbers

- <http://medesign.seas.upenn.edu/index.php/Guides/GeneratingGerberFilesForS62>

Descarga y tutorial de KiCad

- <http://kicad-pcb.org/>
- <https://www.youtube.com/watch?v=13ZixE7oVdg>

Pasar de Eagle a KiCad y librerías

- <http://hackaday.com/2015/12/27/eagle-to-kicad-made-easy/>
- <https://github.com/lachlanA/eagle-to-kicad>

Software Arduino (UnoJoy)

- <https://code.google.com/archive/p/unojoy/>
- <https://github.com/AlanChatham/UnoJoy>

Tutorial para usar Arduino como Joystick

- <http://garagelab.com/profiles/blogs/tutorial-unojoy-utilize-seu-arduino-uno-como-um-joystick>
- <https://dfu-programmer.github.io/>