****

**TREBALL FINAL DE GRAU**

|  |
| --- |
| **TÍTOL DEL TFG: SISTEMA SENSOR**  **TITULACIÓ: Grau en Enginyeria de Sistemes de Telecomunicació**  **AUTOR: Alejandro García Moreno**  **DIRECTOR: Francesc Josep Robert Sanxis**  **DATA: 23 de setembre del 2016** |

Índice

[Capítulo 0. Abstract 4](#_Toc459119243)

[Capítulo 1. Especificaciones del proyecto 5](#_Toc459119244)

[1.1 Lista de especificaciones 5](#_Toc459119245)

[1.2 Teoría general 5](#_Toc459119246)

[1.3 Diagrama general 5](#_Toc459119247)

[Capítulo 2. Planificación 6](#_Toc459119248)

[2.1 Fase 1 del proyecto 6](#_Toc459119249)

[2.1.1 Interrupción INT0IF 6](#_Toc459119250)

[2.1.2 Muestreo del ADC con el Timer3 6](#_Toc459119251)

[2.2 Fase 2 del proyecto 6](#_Toc459119252)

[2.2.1 Comunicación con el sensor DHT22 6](#_Toc459119253)

[2.2.2 Comunicación I2C 6](#_Toc459119254)

[2.3 Fase 3 del proyecto 6](#_Toc459119255)

[2.3.1 PWM 6](#_Toc459119256)

[Capítulo 3. Desarrollo 7](#_Toc459119257)

[3.1 Desarrollo de la Fase 1 7](#_Toc459119258)

[3.1.1 Interrupción INT0IF 7](#_Toc459119259)

[3.1.2 Muestreo del ADC con el Timer3 7](#_Toc459119260)

[3.2 Desarrollo de la Fase 2 7](#_Toc459119261)

[3.2.1 Comunicación con el sensor DHT22 7](#_Toc459119262)

[3.2.2 Comunicación I2C 7](#_Toc459119263)

[3.3 Desarrollo de la Fase 3 7](#_Toc459119264)

[3.3.1 PWM 7](#_Toc459119265)

[Capítulo 4. Implementación del sistema 9](#_Toc459119266)

[4.1 Elección de los sensores 9](#_Toc459119267)

[4.2 Implementación de nuestra aplicación Labview 9](#_Toc459119268)

[4.2.1 Funcionalidades 9](#_Toc459119269)

[4.2.2 Diseño de la aplicación 10](#_Toc459119270)

[4.3 Implementación de sistema sensor en placa de pruebas 11](#_Toc459119271)

[4.3.1 Esquema de conexión de los sensores con el MCU 12](#_Toc459119272)

[4.3.2 Esquema de conexión PICKIT y UART-BLUETOOTH 13](#_Toc459119273)

[4.3.3 Esquema de alimentación de la Development Board 14](#_Toc459119274)

[4.4 Implementación del sistema sensor en PCB con Eagle 16](#_Toc459119275)

[4.4.1 Schematic 16](#_Toc459119276)

[4.4.2 Board 17](#_Toc459119277)

[4.4.3 Mecanizado 19](#_Toc459119278)

[Capítulo 5. Conclusiones 21](#_Toc459119279)

# Abstract

El proyecto trata de implementar un sistema de control de riego y lumínico para un invernadero. Para realizar dicho control, el sistema deberá de medir con sensores diferentes paramentos como la temperatura ambiente, la humedad relativa del aire, la humedad de la tierra y la intensidad lumínica. Una vez medidos estos valores, el sistema sensor deberá de enviar las ordenes correspondientes a los diferentes actuadores y además enviar, vía Wireless, esos datos a nuestra aplicación Labview para ser mostrados por pantalla y poder realizar un estudio de estos parámetros en base al tiempo.

El sistema sensor se tendrá que realizar con un microcontrolador PIC18F de bajo consumo para que el sistema pueda ser autónomo con poca energía de alimentación. El PIC18F será programado en C usando el estilo de programación FSM. Los primeros programas se simularán en el simulador virtual Proteus para poder detectar bugs de una manera más eficaz. Una vez realizadas las simulaciones, el programa será cargado en nuestro PIC físico en nuestra placa de pruebas donde comprobaremos que todo lo simulado funciona.

En la elección de sensores, se intentará trabajar con sensores analógicos y digitales indistintamente. Respecto los sensores con interface digital, será importante realizar un estudio de sus protocolos de comunicación.

Finalmente, después de realizar el prototipo final en nuestra placa de pruebas, se diseñará con Eagle la correspondiente placa de circuito impreso(PCB) de nuestro sistema sensor y ,a poder ser, también se hará el mecanizado de la misma.

# Especificaciones del proyecto

## Lista de especificaciones

Especificaciones del sistema

Medir magnitudes fisicas

Micro controlador de xlp

Autonomo

Aplicación de monotorización

Control de actuadores via MCU

## Teoría general

Como medir las magnitudes fisicas Sensores 1

Modulos inalambricos

Energy

valvulas

## Diagrama general

Diagrama del sistema

dibujo

# Planificación

Estructura del programa principal

## Fase 1 del proyecto

### Interrupción INT0IF

### Muestreo del ADC con el Timer3

Creación de un timer y interrupción por boton ADC

## Fase 2 del proyecto

### Comunicación con el sensor DHT22

### Comunicación I2C

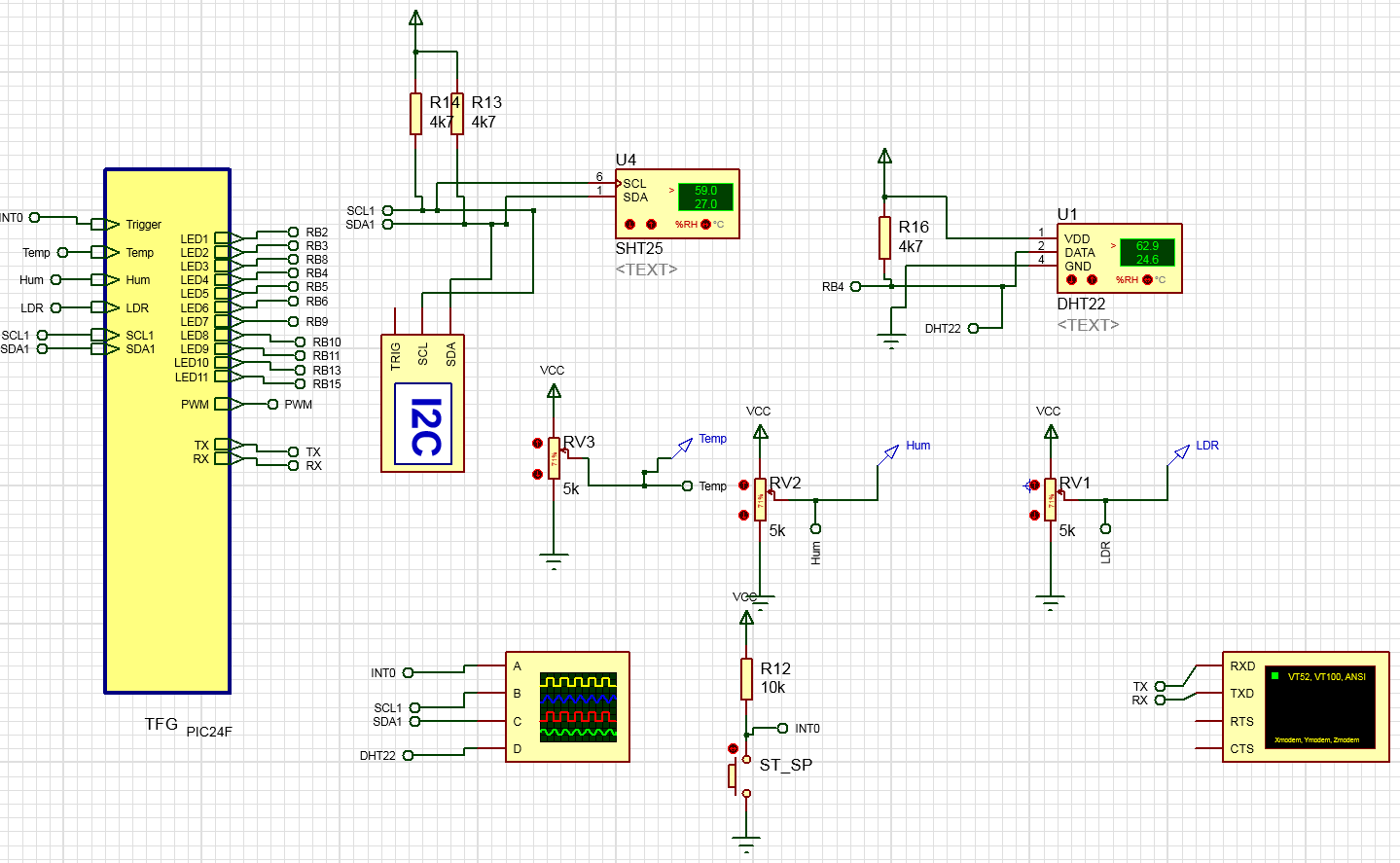
## Fase 3 del proyecto

### PWM

comunicación con sensores digitales

# Desarrollo

Para realizar el desarrollo de las diferentes fases de planificación hemos utilizado el simulador Proteus. El simulador Proteus nos permite simular nuestro PIC18F26K20, algunos sensores digitales, la comunicación UART y otros elementos que nos han ayudado a probar el ADC y la comunicación I2C. Además, Proteus nos permite realizar debugging y de esta manera poder encontrar rápidamente soluciones a un bugs en el código.



**Fig. 3.1** Esquemático de nuestro proyecto Proteus

## Desarrollo de la Fase 1

### Interrupción INT0IF

### Muestreo del ADC con el Timer3

## Desarrollo de la Fase 2

### Comunicación con el sensor DHT22

### Comunicación I2C

## Desarrollo de la Fase 3

### PWM

pwm acuador

# Implementación del sistema

## Elección de los sensores

## Implementación de nuestra aplicación Labview

Desde nuestra aplicación de Labview podremos recibir los datos y transmitir información por un puerto COM.

### Funcionalidades

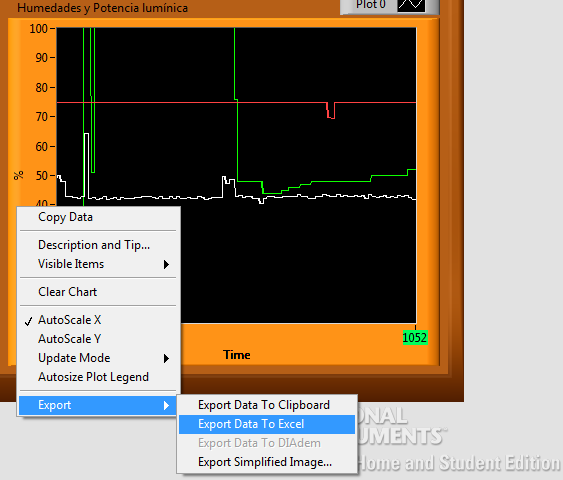
La comunicación aplicación-MCU es bidireccional, desde el Panel de control podemos sacar del modo Sleep a nuestra MCU y que nos envíe los datos de forma instantánea.

Nuestra aplicación realiza un parsing de los mensajes que nos envía el sistema sensor para extraer los datos. Los datos de temperatura los mostramos en los termómetros y los demás datos en medidores en forma de porcentaje.



**Fig. 6.1** Interfaz gráfica de nuestra aplicación Labview

Además, cada dato de los sensores pueden ser dibujados en las gráficas y también podemos exportar las gráficas a Excel.

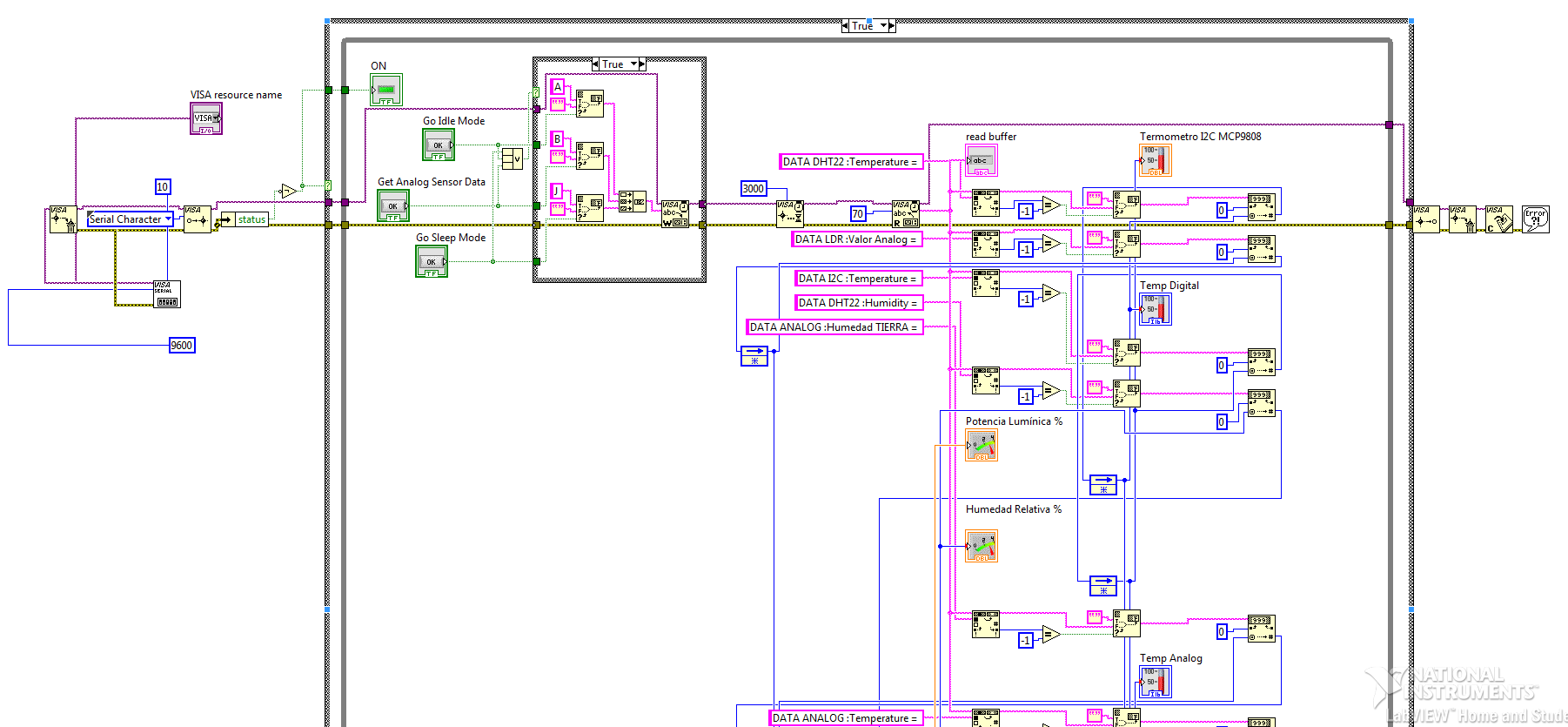


**Fig. 6.2** Exportación de los datos en excel

### Diseño de la aplicación

Labview es una plataforma de diseño de aplicaciones de ingeniería y ciencia pensada para acelerar la productividad de los ingenieros.  Con una sintaxis de programación gráfica que facilita visualizar, crear y codificar sistemas de ingeniería.( http://www.ni.com/labview/esa/)

La creación de la aplicación consiste en ir uniendo módulos y al unirlos formar un diagrama de bloques.(Ver esquemático)

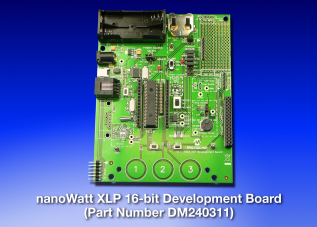


**Fig. 6.3** Esquemático de nuestra aplicación en desarrollo

[Ver explicación detallada de la aplicación ANEXO]

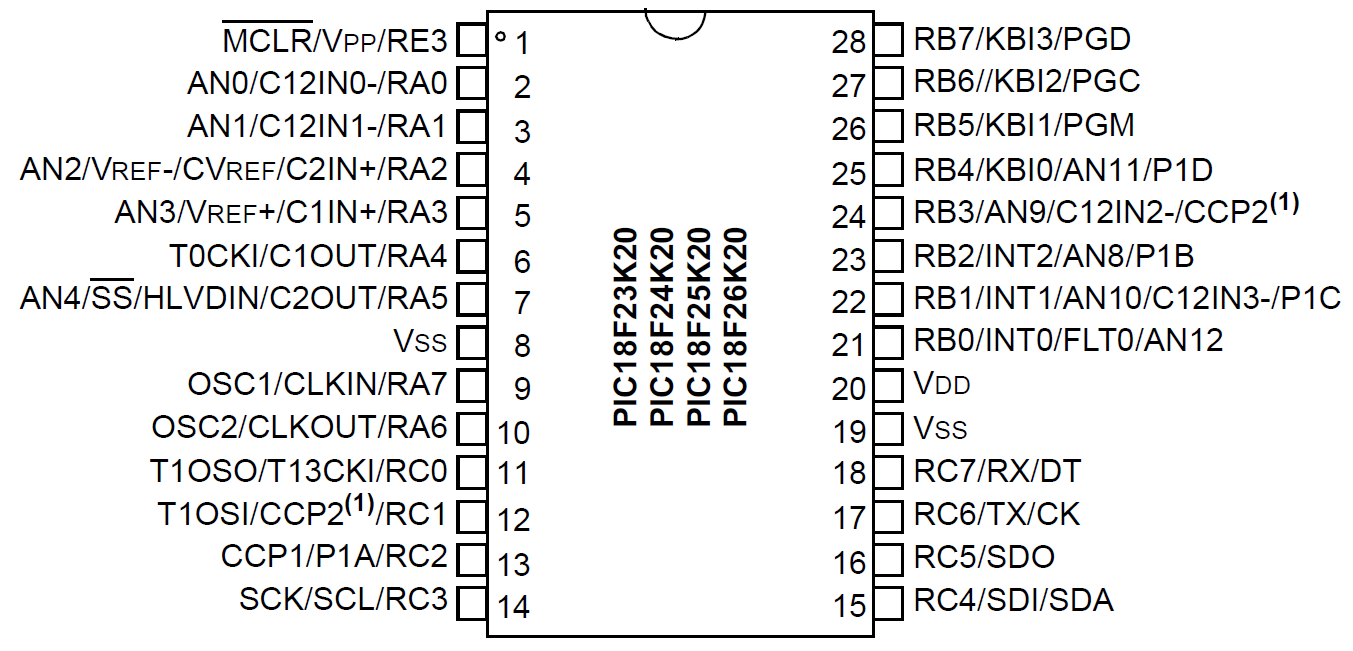
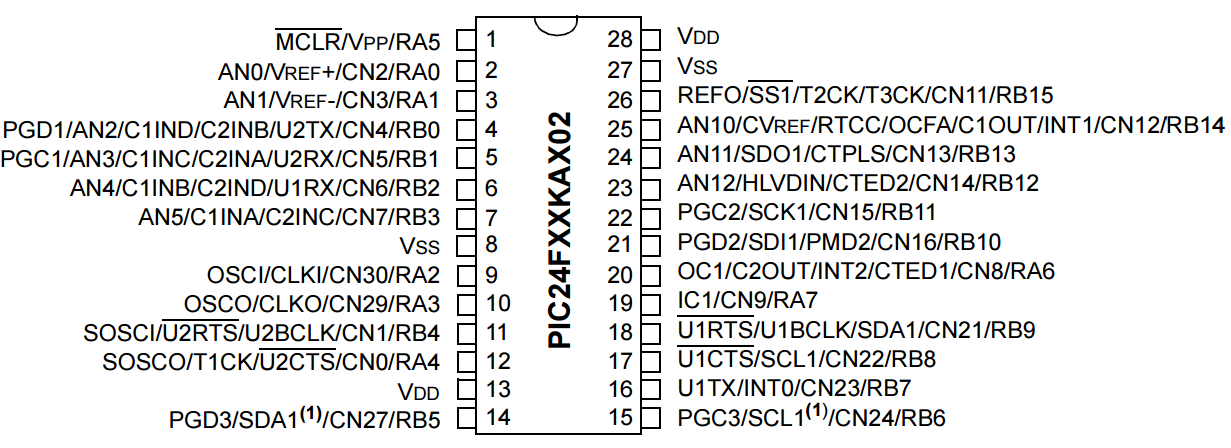
## Implementación de sistema sensor en placa de pruebas

En los inicios del desarrollo del sistema sensor, utilizamos la placa de entrenamiento XLP 16-bits Development Board de Microchip. El mayor inconveniente de trabajar con esta placa era conectar nuestro PIC18 porque la placa está diseñada para un PIC24F16KA102.



**Fig. 7.1** XLP 16-bits Development Board de Microchip

La solución a este problema fue trabajar con una protoboard para poder conectar las I/O del PIC18 con la placa de manera correcta.



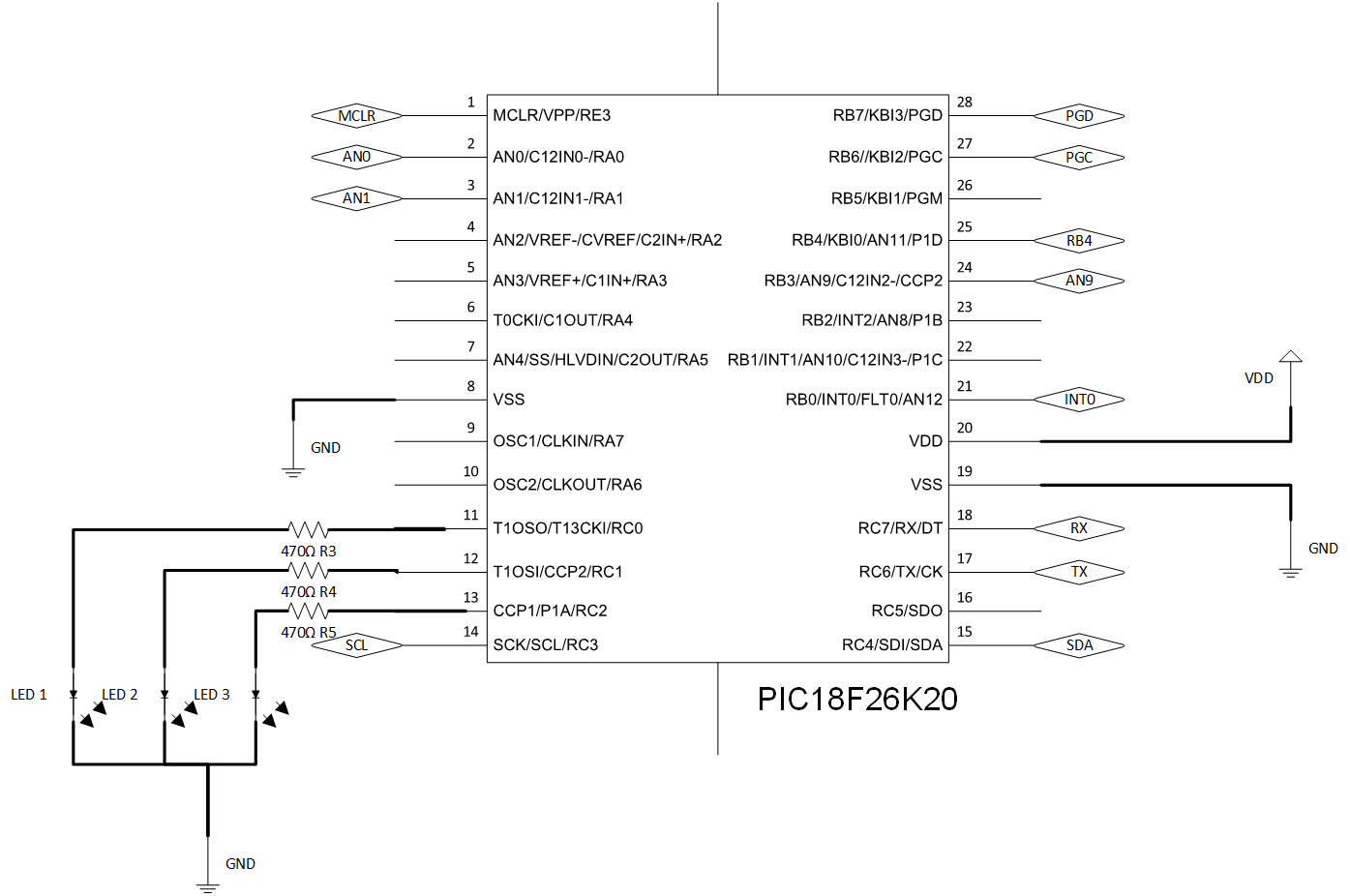
**Fig. 7.2** Diagrama de pines: arriba PIC24F16KA102 y abajo nuestro PIC18F26K20

De este modo fue posible utilizar los dispositivos de la placa de entrenamiento, los más importantes: botones, conector pickit, potenciómetro y circuito de alimentación.



**Fig. 7.3** Implementación del sistema sensor en placa de pruebas

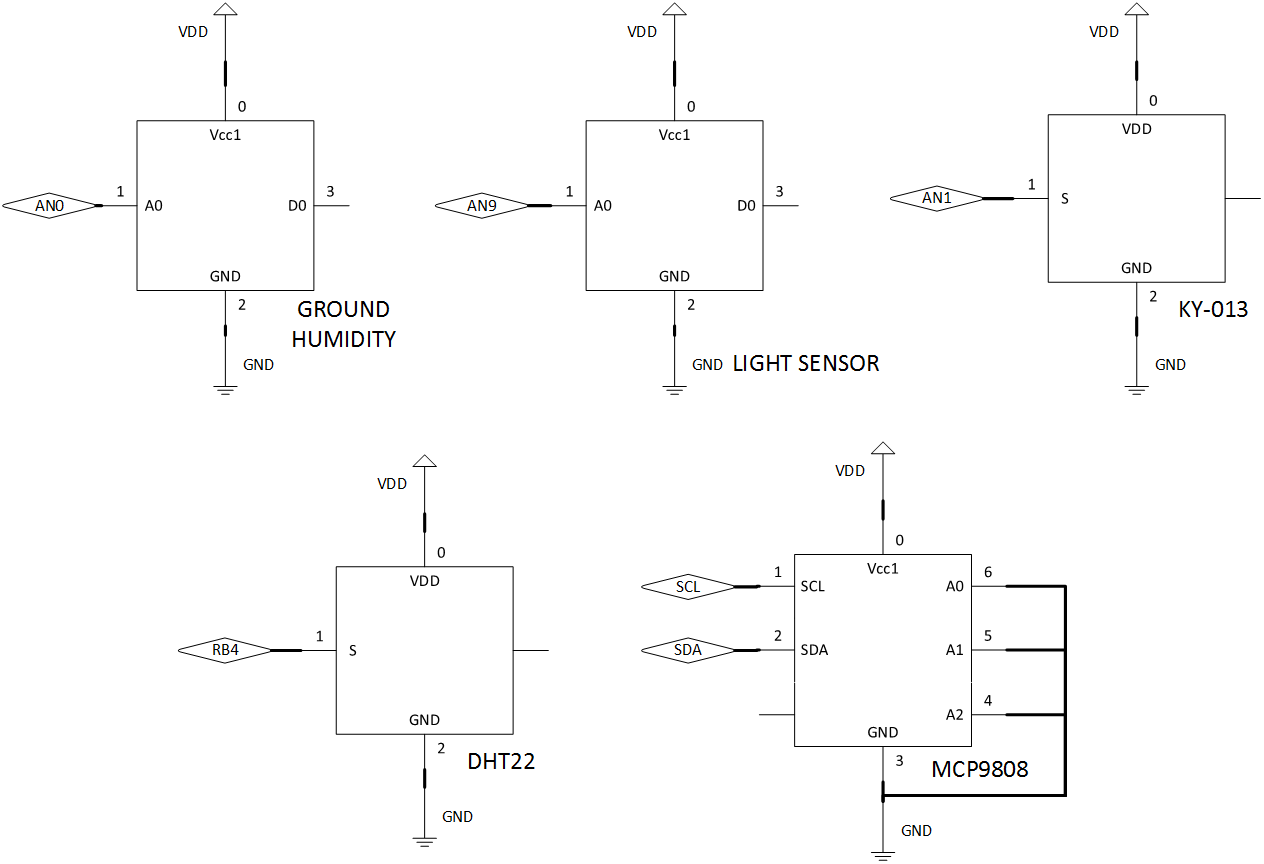
### Esquema de conexión de los sensores con el MCU



**Fig. 7.4** Conexiones con nuestro MCU

Además de conectar los sensores al MCU, hemos conectado unos LEDs en los puertos de salida RC0, RC1, RC2 para tener una referencia del estado en el que se encontraba nuestra máquina de estados.

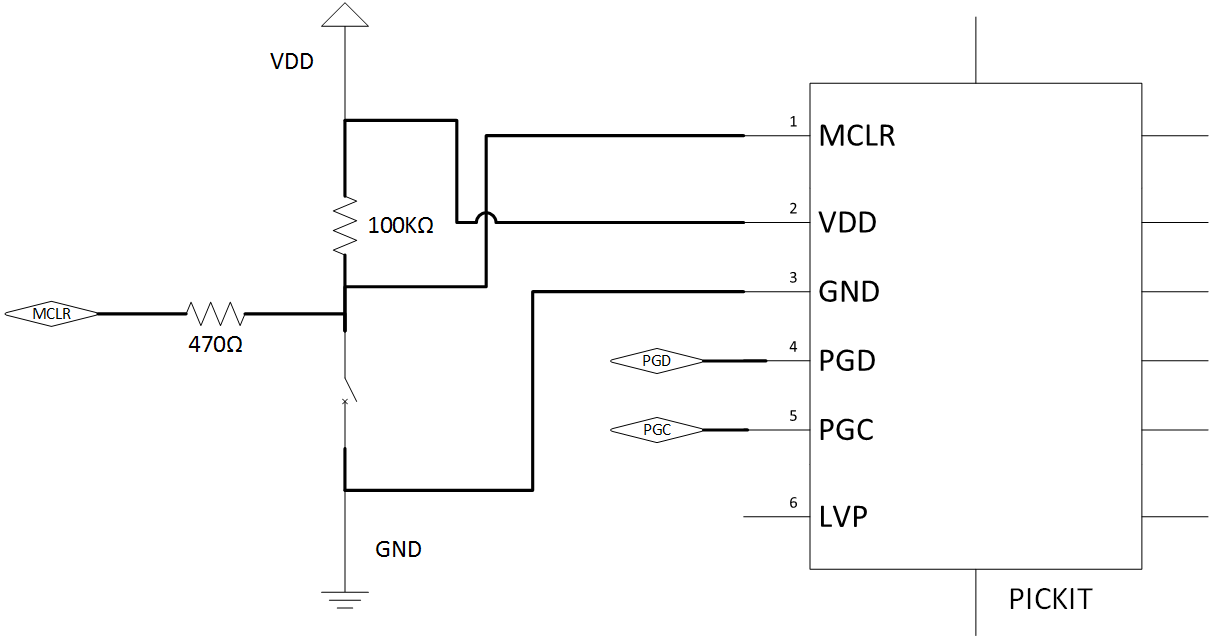
Las conexiones de los sensores se han realizado como explicamos en apartado 4 sin embargo no hemos puesto las resistencias de pull up porque ya vienen integradas en la PCBs que integran los sensores[Anexos].



**Fig. 7.5** Conexiones de nuestros sensores con nuestro MCU

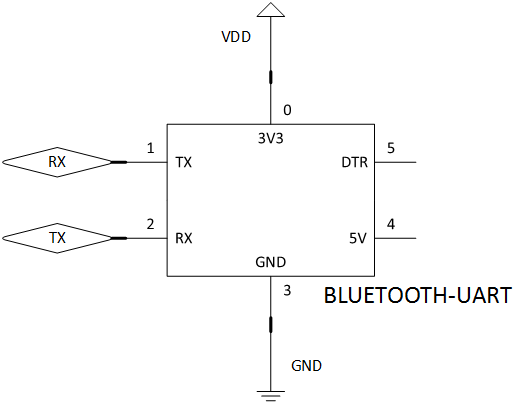
### Esquema de conexión PICKIT y UART-BLUETOOTH

Para programar nuestro PIC es necesario tener acondicionado los pines del PICKIT siguiendo el esquema que nos indica Microchip. Utilizando la placa de entrenamiento no tendremos problema porque ya viene integrada la conexión con PICKIT pero si tuviéramos que conectarlo sin la placa de entrenamiento deberíamos de realizar las conexiones siguiendo el siguiente esquema[Ver Esquema y Anexo PICKIT].



**Fig. 7.6** Conexión con Pickit

Cuando conectemos nuestro transmisor-receptor bluetooth a nuestro MCU, deberemos conectar el puerto RXmcu al TXbluetooth y el TXmcu al RXbluetooth[Ver].

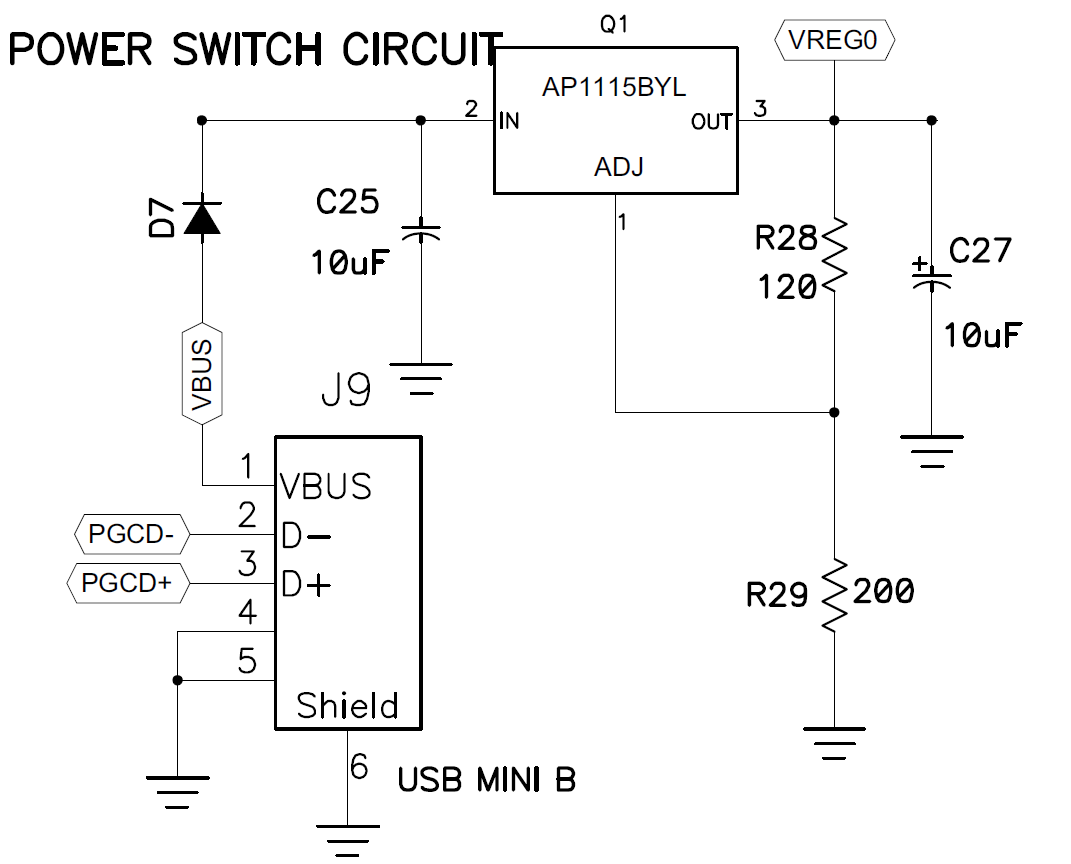


**Fig. 7.7** Conexiones de nuestro módulo bluetooth con nuestro MCU

### Esquema de alimentación de la Development Board

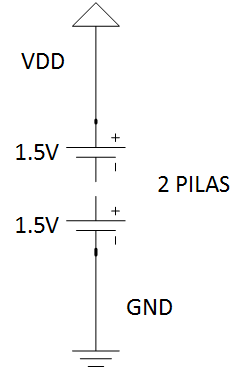
Hay 4 maneras de alimentar nuestro sistemas sensor con nuestra placa de entrenamiento, una utilizar el esquema de conexión del PICKIT y hacer que alimente el PICKIT todo nuestro sistema.

La segunda opción es utilizar la entrada USBmini y con el siguiente sistema regulador convertir los 5V del USB a 3.3V(VDD de nuestro prototipo).[Ver esquema, explicación Anexos]



**Fig. 7.8** Circuito de alimentación por mini USB

La tercera opción es utilizar las pilas de mercurio para realizar una alimentación de 2,9-3V. Este modo de conexión no es el más óptimo porque el voltaje de alimentación es inferior al voltaje de funcionamiento de alguno de nuestros sensores. Aún teniendo este inconveniente el sistema sensor excepto el sensor DHT22 que no funciona.[Ver Esquema]



**Fig. 7.9** Circuito de alimentación con 2 pilas de 1.5V

La cuarta opción consiste en utilizar el Cymbet’s EVAL-08 Solar Energy Harvester que incluye nuestra development board. El mismo está diseñado para ir cargando poco a poco una batería y que el sistema consuma nA, sin embargo nuestro sistema necesita 3.3V de alimentación constante por lo tanto no se puede utilizar para alimentar el sistema sensor.



**Fig. 7.10** Alimentación con Cymbet’s EVAL-08 Solar Energy Harvester

## Implementación del sistema sensor en PCB con Eagle

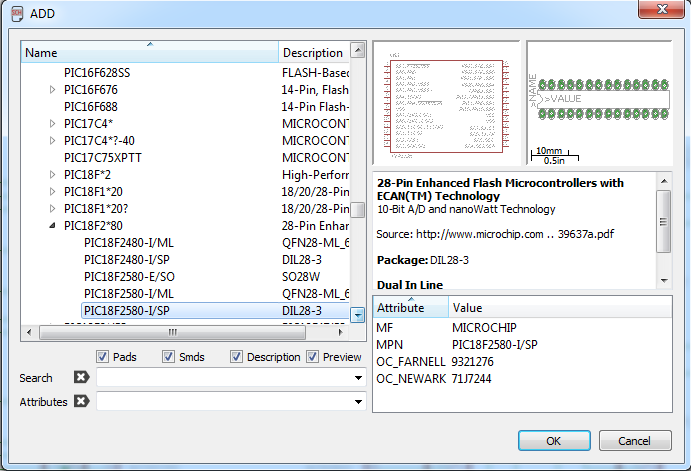
EAGLE, (siglas de Easily Applicable Graphical Layout Editor) es un programa de diseño de diagramas y PCBs con autoenrutador.

1. Lo primero que deberemos realizar para el diseño de nuestra propia PCB, es dibujar el esquema eléctrico de lo que va a ser nuestra placa para luego generar nuestra Broad.
2. Una vez generada la board, deberemos de colocar los elementos del circuito en nuestra placa e ir conectando los elementos.
3. Una vez acabado del diseño de la placa, generaremos los archivos gerber para realizar el mecanizado de nuestra placa.

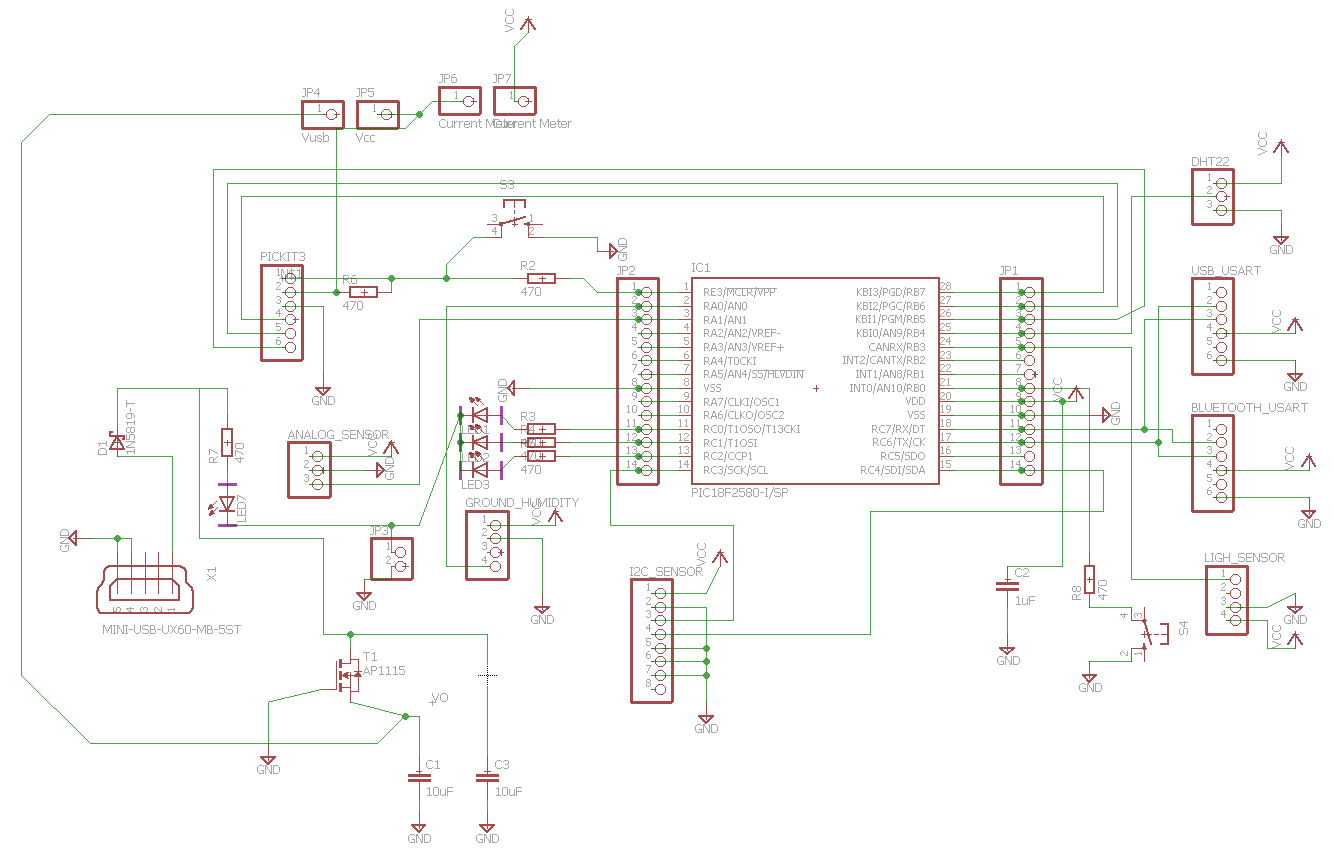
### Schematic

Lo primero que hay que hacer antes de empezar, es saber qué package tienen nuestros elementos del circuito porque el package delimita qué agujeros y/o forma de la pista para luego el dispositivo encaje al soldarlo. Dicho esto, en las librerías de Eagle no tienen todos los dispositivos pero casi todos los tipos de package sí suelen estar. Entonces, cuando no encoramos el dispositivo podemos hacer 2 cosas: crear nuestro propio dispositivo utilizando un package de la librería o coger un dispositivo existente con el mismo package.

En nuestro diseño hemos cogido elementos existentes con el mismo package para hacer el esquemático porque lo importante es el diseño de la board.

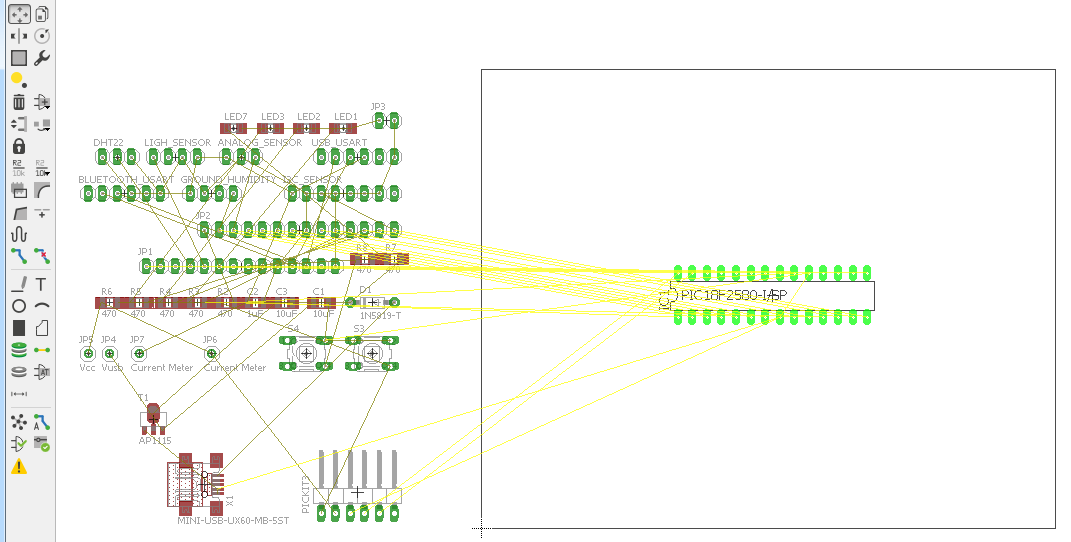


El esquema es casi el mismo que en la placa de pruebas, la única diferencia la tenemos en la alimentación[Anexos], y, pensando en la board, hemos cambiado la mayoría de los package a SMD categoria 1206.[Detalles en anexos] Las conexiones con los sensores y bluetooth se han hecho con puertos jumpers hembra.

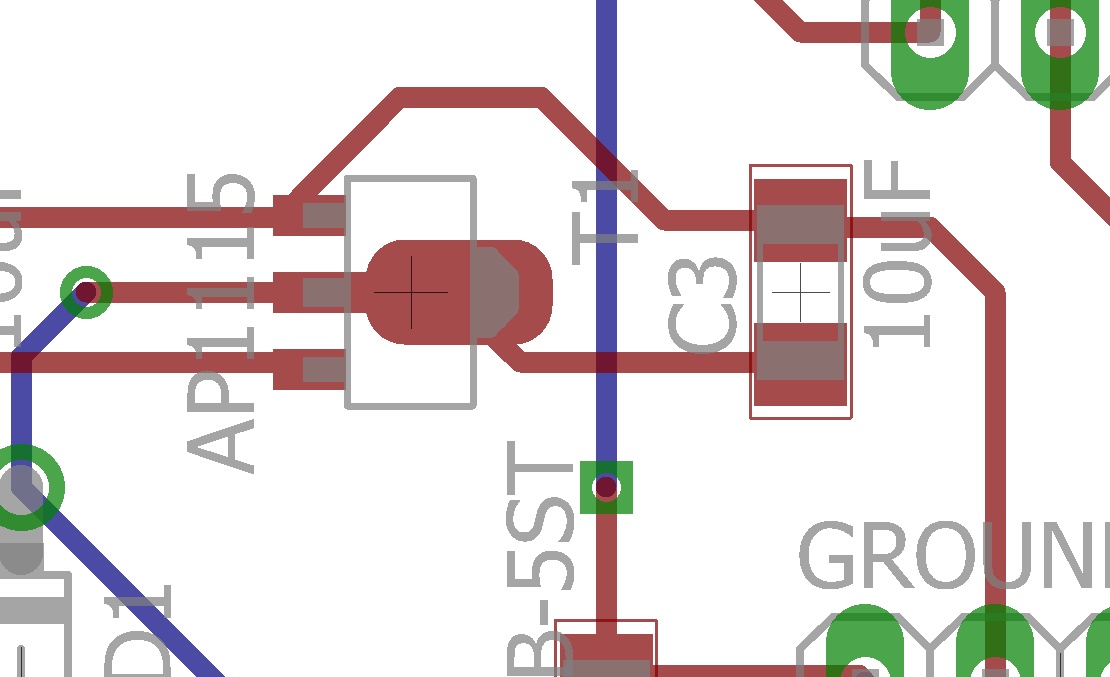


### Board

Una vez realizado el esquematizo, tenemos que generar nuestra board. Con la board generada, tenemos que ir colocando los elementos pensado cómo las pistas tendrán que conectar entre pin y pin.

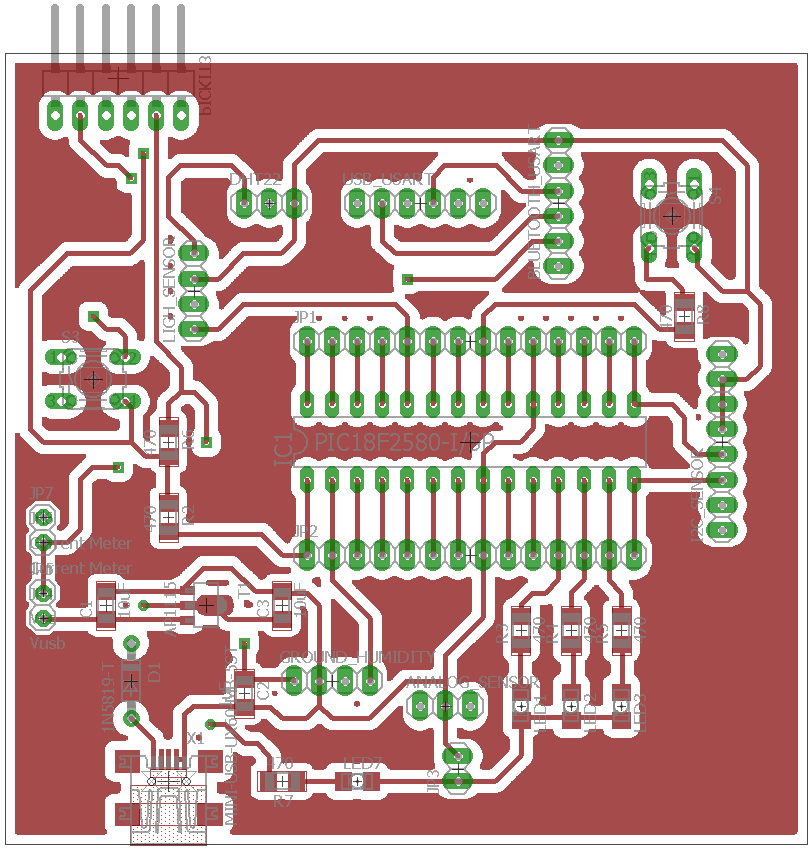


Con todos los elementos colocados, tenemos que dibujar las pistas sin que se solapen entre ellas. Para dibujar las pistas, podemos usar el autorute pero la mejor opción es dibujarlas de forma manual porque normalmente el autorute dibuja rutas incoherentes y dificulta el cumplimiento *design rules*. Nuestra placa es de doble capa(TOP y BOTTOM), entonces cuando dibujamos las pistas podemos usar sin ningún problema vías para pasar una pista por debajo de otra para evitar que se solapen[Ver ].

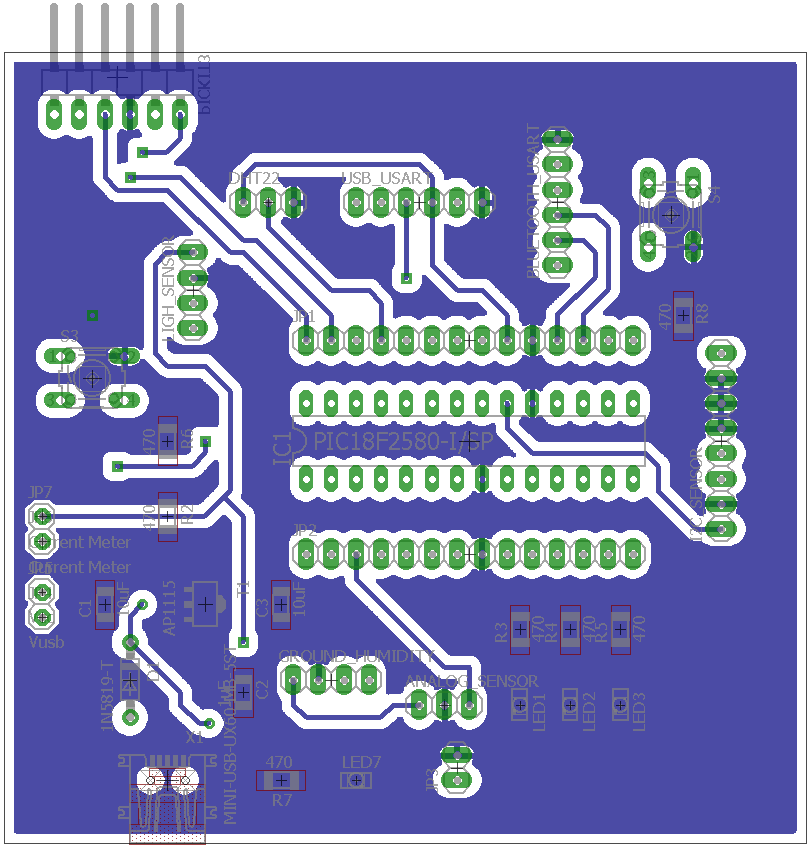


(línea roja capa top y azul bottom)

Finalmente, tenemos que dibujar un polígono que ocupe todo el área de la placa para que Eagle nos dibuje el cobre que no se utiliza que está al lado de las pistas.[Ver Cara Top][Anexos]



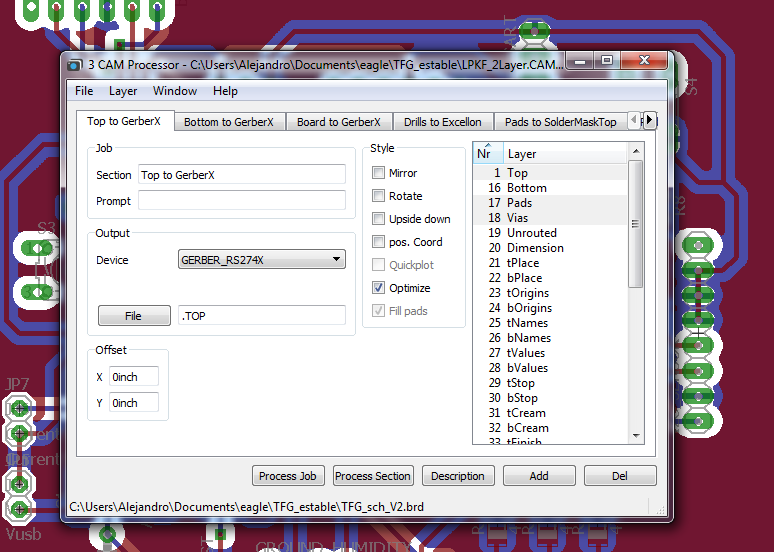
Además en la cara bottom conectaremos el cobre sobrante a masa[Ver cara bottom]



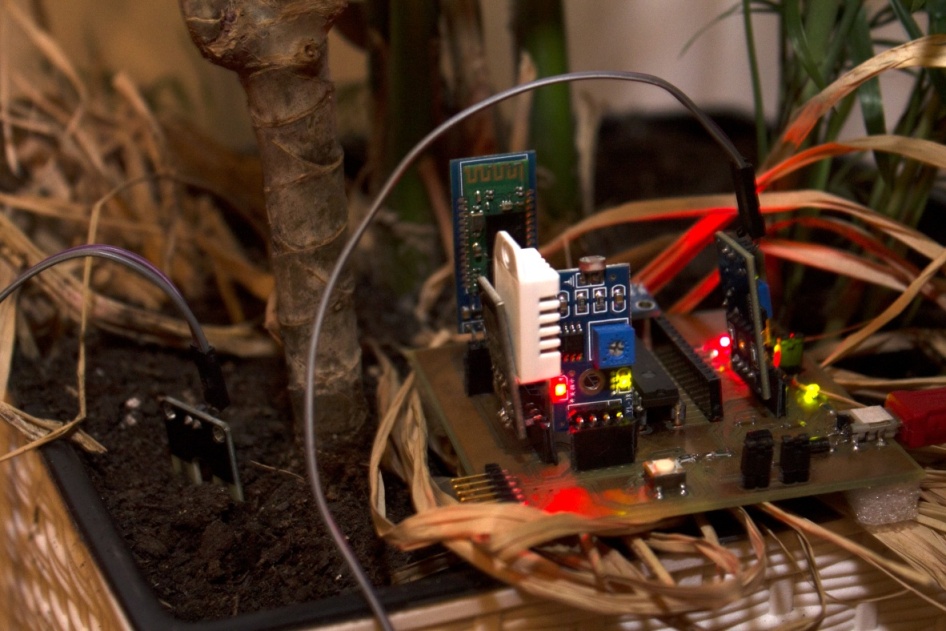
(Cara bottom-plano de masa)

### Mecanizado

Con la placa acabada, sólo falta generar los archivos gerber para realizar el mecanizado de nuestra placa. Deberemos de generar 4 archivos: TOP, BOTTOM, DRILLS y BOARD.



El proceso de mecanizado queda explicado en los anexos.[Anexos]

****

(PCB funcionando)

# Conclusiones