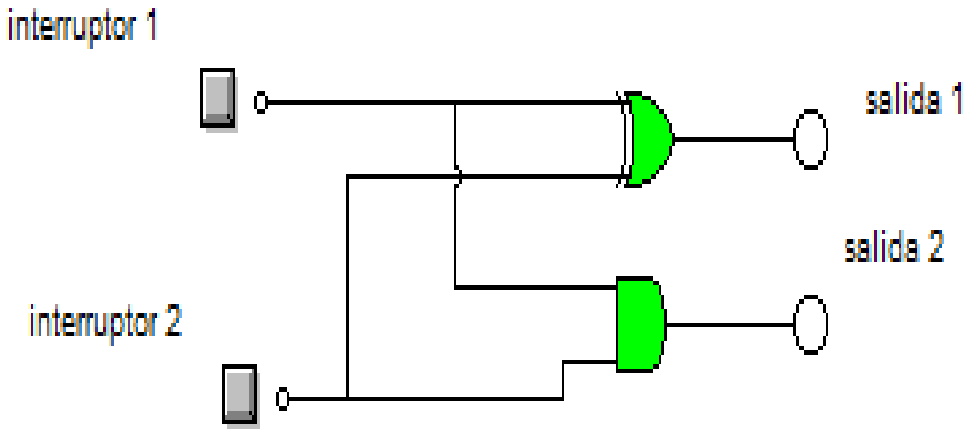
SUMADOR

En electrónica un sumador es un circuito lógico que calcula la operación suma. En los computadores modernos se encuentra en lo que se denomina Unidad aritmético lógica. Generalmente realizan las operaciones aritméticas en código binario decimal o BCD exceso 3, por regla general los sumadores emplean el sistema binario. En los casos en los que se esté empleando un complemento a dos para representar números negativos el sumador se convertirá en un sumador-restador.

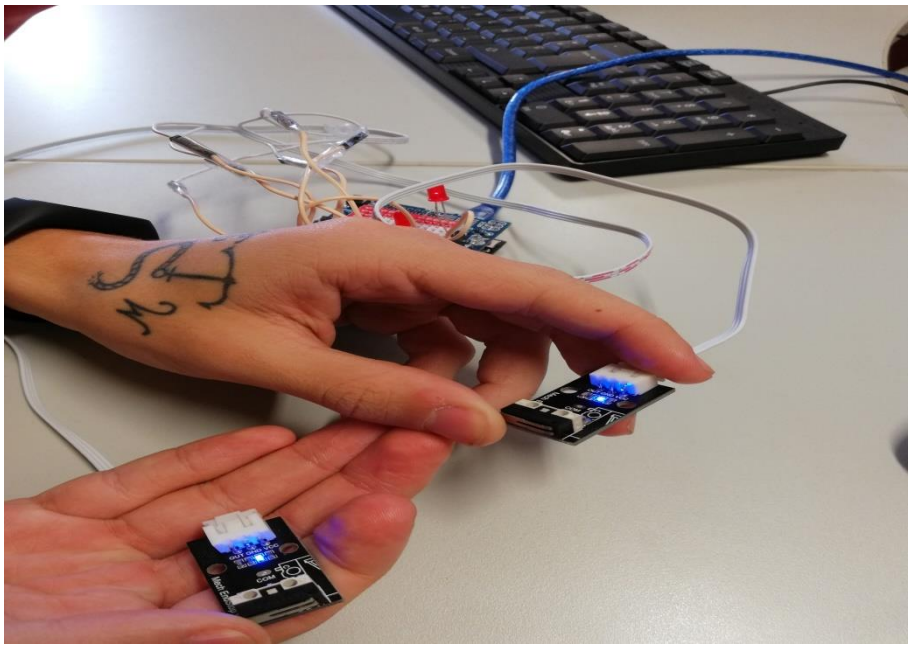


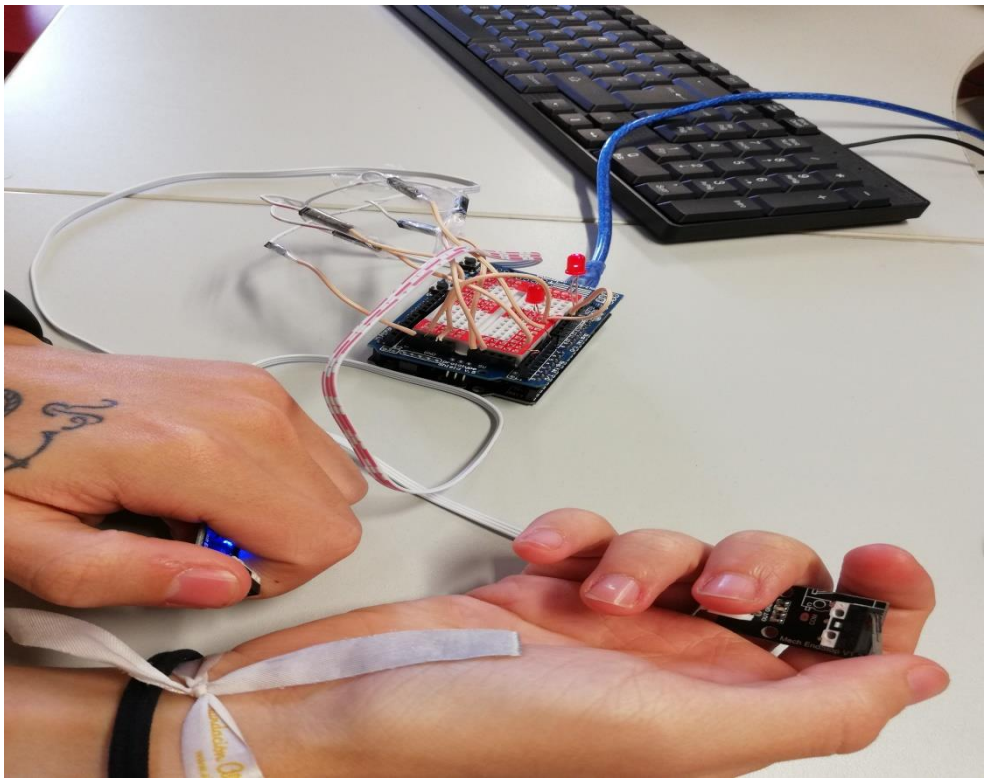
A	B	S1	S2
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

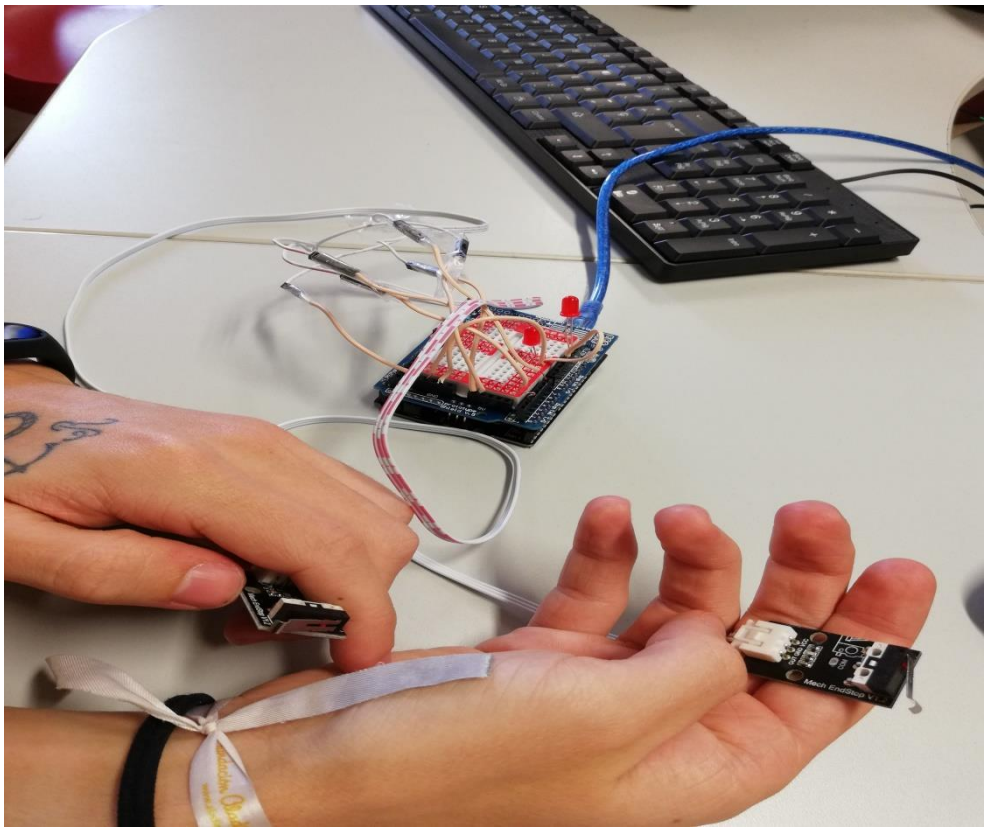
CÓDIGO:

```
/*
  Boole
  Función AND con 2 variables
 */

int var1 = 7;   //Pin de entrada del pulsador 1
int var2 = 2;   //Pin de entrada del pulsador 2
int led = 5;    //Pin de salida para el led(rojo)
int estado1 = 0;    //Para almacenar el estado de la variable1
int estado2 = 0;    //Para almacenar el estado de la variable2
int resultado = 0;  //Para almacenar el resultado

void setup() {
  pinMode(var1, INPUT);     //Iniciliza el pin de entrada 1 como salida
  pinMode(var2, INPUT);     //Iniciliza el pin de entrada 2 como salida
  pinMode(led, OUTPUT);     //Iniciliza el pin del led como salida
}

void loop(){
  estado1 = digitalRead(var1);  //Lee el estado del botón y lo almacena
  estado2 = digitalRead(var2);  //Lee el estado del botón y lo almacena
  resultado = (estado1 && estado2); //Función AND con los dos estados
  digitalWrite(led, resultado);    //Escribimos el resultado en el led
}
```

fotos arduino:

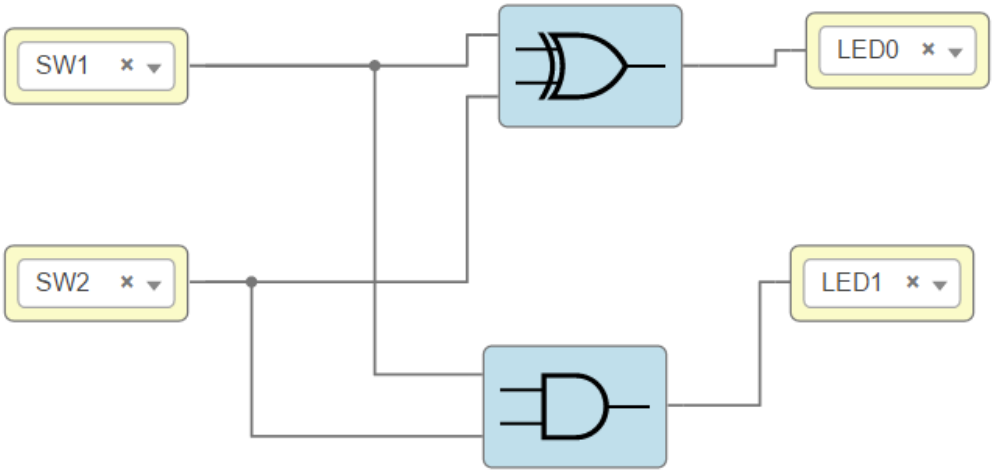




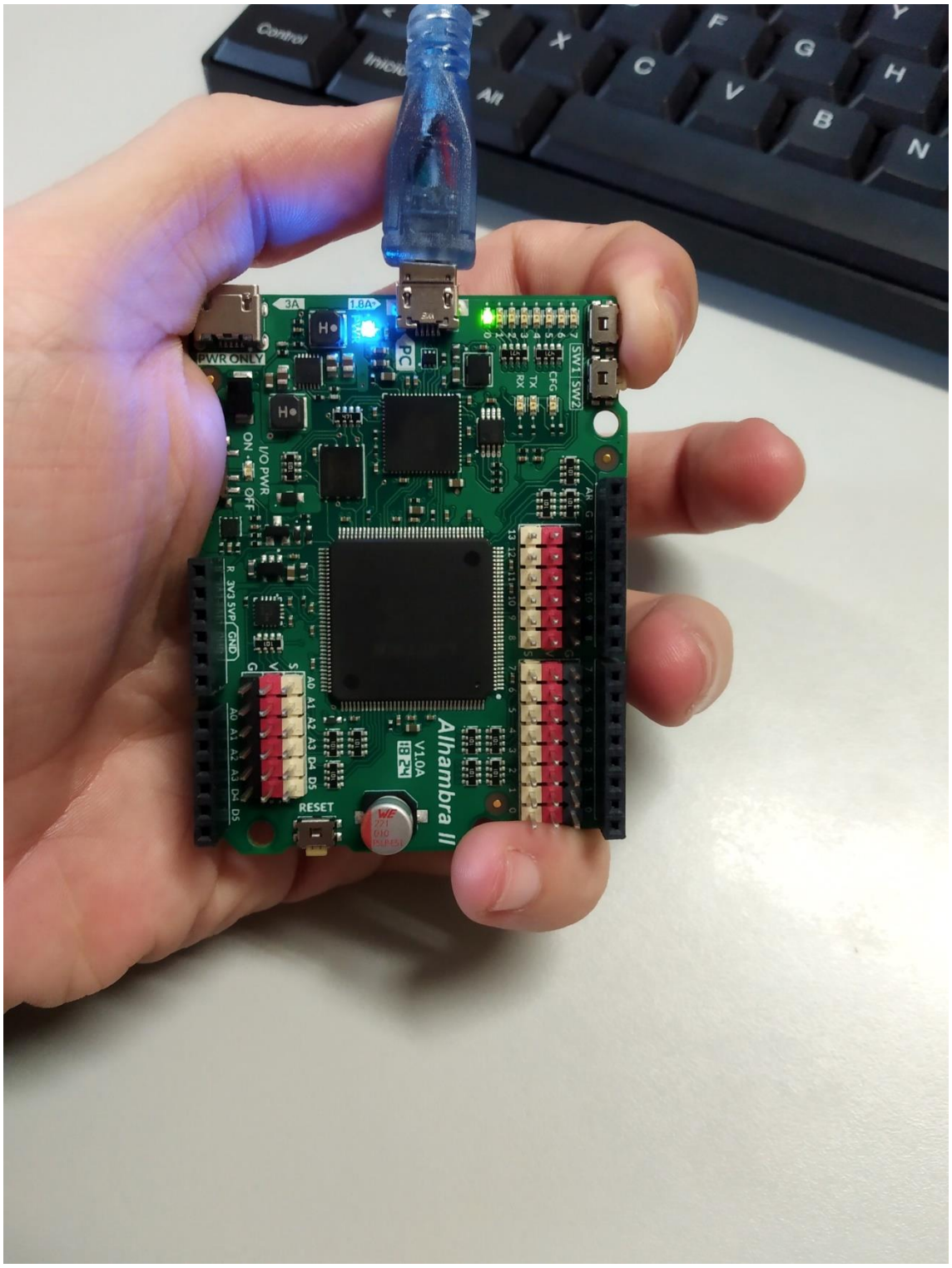


```
int var1 = 7;   //Pin de entrada del pulsador 1
int var2 = 2;   //Pin de entrada del pulsador 2
int led1 = 13;    //Pin de salida para el led(rojo)
int led2 = 8;    //Pin de salida para el led(rojo)

int estado1 = 0;    //Para almacenar el estado de la variable1
int estado2 = 0;    //Para almacenar el estado de la variable2
int resultado1 = 0;    //Para almacenar el estado de la variable1
int resultado2 = 0;    //Para almacenar el estado de la variable2

void setup() {
  pinMode(var1, INPUT);     //Iniciliza el pin de entrada 1 como salida
  pinMode(var2, INPUT);     //Iniciliza el pin de entrada 2 como salida
     //Iniciliza el pin del led como salida
   pinMode(led1, OUTPUT);     //Iniciliza el pin del led como salida
    pinMode(led2, OUTPUT);     //Iniciliza el pin del led como salida
}

void loop(){
  estado1 = !digitalRead(var1);  //Lee el estado del botón y lo almacena
  estado2 = !digitalRead(var2);  //Lee el estado del botón y lo almacena
  resultado1 = (estado1 and estado2); //Función AND con los dos estados
  digitalWrite(led1, resultado1);    //Escribimos el resultado en el led

 resultado2 = (estado1 xor estado2); //Función AND con los dos estados
  digitalWrite(led2, resultado2);    //Escribimos el resultado en el led
```

FPGA



Fotos FPGA:


Control
Alt
PWR ONLY
3A
1.8A
PC
SW1
SW2
CFG
TX
RX
I/O PWR
ON
OFF
3V3
GND
RESET
Alhambra II
V1.0A


Alhambra II
V1.0A
RESET
PWR ONLY
I/O PWR
ON
OFF
SW1
SW2
PC
3A
1.8A


3A
1.8A
PWR ONLY
I/O PWR
ON
OFF
SW1
SW2
CFG
TX
RX
3V3
5V0
GND
RESET
V1.0A
Alhambra II

DIFERENCIA ARUDINO FPGA

ARDUINO	FPGA
es una compañía de desarrollo de hardware y software, compuesta respectivamente por circuitos impresos que integran un microcontrolador y un entorno de desarrollo (IDE)	Una FPGA (del inglés Field Programmable Gate Array) es un dispositivo programable que contiene bloques de lógica cuya interconexión y funcionalidad puede ser configurada 'in situ' mediante un lenguaje de descripción especializado.
El hardware consiste en una placa de circuito impreso con un microcontrolador, usualmente Atmel AVR, puertos digitales y analógicos de entrada/salida,[4] los cuales pueden conectarse a placas de expansión (shields), que amplían las características de funcionamiento de la placa Arduino. Asimismo, posee un puerto de conexión USB desde donde se puede alimentar la placa y establecer comunicación con el computador	Se utilizan en aplicaciones similares a los ASICs sin embargo son más lentas, tienen un mayor consumo de energía y no pueden abarcar sistemas tan complejos como ellos. A pesar de esto, las FPGAs tienen las ventajas de ser reprogramables (lo que añade una enorme flexibilidad al flujo de diseño), sus costes de desarrollo y adquisición son mucho menores para pequeñas cantidades de dispositivos y el tiempo de desarrollo es también menor.
Barato: Las placas Arduino son relativamente baratas comparadas con otras plataformas microcontroladoras. La versión menos cara del modulo Arduino puede ser ensamblada a mano, e incluso los módulos de Arduino preensamblados cuestan menos de 50$	Cada chip de FPGA está hecho de un número limitado de recursos predefinidos con interconexiones programables para implementar un circuito digital reconfigurable y bloques de E/S para permitir que los circuitos tengan acceso al mundo exterior.
Entorno de programación simple y claro: El entorno de programación de Arduino es fácil de usar para principiantes, pero suficientemente flexible para que usuarios avanzados puedan aprovecharlo también. Para profesores, está convenientemente basado en el entorno de programación Processing, de manera que estudiantes aprendiendo a programar en ese entorno estarán familiarizados con el aspecto y la imagen de Arduino.	la tecnología de FPGA estaba disponible solamente para ingenieros con un profundo conocimiento del diseño de hardware digital. El surgimiento de herramientas de diseño de alto nivel, como NI LabVIEW, cambia las reglas de programación de FPGAs, ofreciendo nuevas tecnologías que convierten los diagramas de bloques gráficos en circuitos de hardware digital
Código abierto software harware extensible: El software Arduino está publicado como herramientas de código abierto, disponible para extensión por programadores experimentados. El lenguaje puede ser expandido mediante librerías C++, y la gente que quiera entender los detalles técnicos pueden hacer el salto desde Arduino a la programación en lenguaje AVR C en el cual está basado. De forma similar, puedes añadir código AVR-C directamente en tus programas Arduino si quieres.	Uno de los beneficios de los FPGAs ante los sistemas basados en procesador es que la lógica de aplicación es implementada en circuitos de hardware en lugar de ejecutarse aparte de un SO, controladores y software de aplicación.