UNIVERSIDAD NACIONAL DE AVELLANEDA

CARRERA: INGENIERÍA EN INFORMÁTICA

MATERIA: Circuitos y mediciones electrónicas

Primer cuatrimestre del 2022

Trabajo Practico Final: Semáforo inteligente

Datos de los integrantes:

Apellido y Nombres	**E-mails**
López Vidueiros Nicolas	nicolas.lopez.vidueiros@gmail.com
Rodriguez Gisela Solange	gisela.solange.rodriguez@gmail.com

Profesor: Pablo Lannes

Índice

Trabajo de laboratorio: Semáforo inteligente

Objetivo

El objetivo del siguiente proyecto es realizar una representación en escala del funcionamiento de un semáforo inteligente. En este caso se hará uso de dos semáforos, los cuales tendrán la cualidad de cambiar de estado dependiendo si el pulsador es presionado por un peatón o si detecta que se aproxima un vehículo. Para llevar a cabo este trabajo, se utilizarán los conocimientos aprendidos y estudiados en la materia Circuitos y Mediciones Electrónicas.

Implementación

Componentes utilizados:

- Arduino Mega 2560 x1
- Transistores BC547 x6
- Sensores ultrasónicos HC-SR04 x2
- Pulsadores x2
- Leds x6
- Resistencias
    - 150Ω x6
    - 6,8kΩ x6
    - 10kΩ x2
- Cables varios
- Protoboard x1

A continuación, se describirán los componentes utilizados para la realización del proyecto.

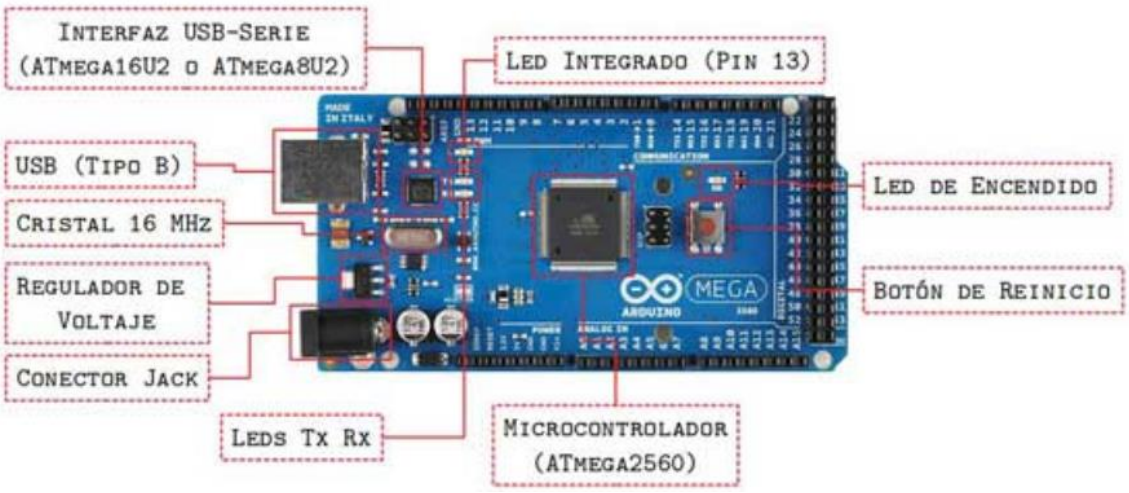
Arduino Mega 2560

Arduino MEGA 2560 es una placa de desarrollo basada en el microcontrolador ATmega2560 (de aquí su nombre). Esta placa pertenece a la extensa familia de placas Arduino, siendo junto al Arduino UNO de las más representativas.

Como es costumbre en esta familia de placas, el Arduino MEGA 2560 está compuesto, básicamente, por:

- Un microcontrolador (ATmega2560) con la configuración de "sistema mínimo" (El término "sistema mínimo" se refiere a que solo se utilizan los componentes indispensables para el microcontrolador).
- Una interfaz USB-Serie que permite re-programar dicho microcontrolador utilizando simplemente un ordenador, un cable USB y el software Arduino IDE.
- Y un conjunto de cabezales que permiten conectar los pines de entrada/salida, ya sea con los conocidos shields o con cualquier otro sistema externo.

A continuación, veremos los principales componentes que constituyen al Arduino MEGA 2560, así como la distribución de los pines.



Microprocesador: podemos describir las siguientes características principales:

- 256 kB de memoria FLASH.
- 8 kB de memoria SRAM.
- 4 kB de memoria EEPROM.
- Frecuencia de CPU Máxima: 16MHz.
- Voltaje de Operación máximo: 6.0V.

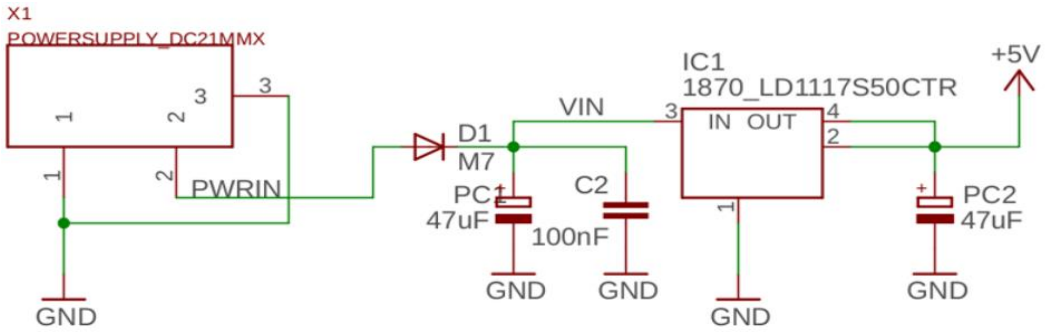
Conector USB: podemos utilizarlos para las siguientes dos funciones:

- Alimentar la placa utilizando los 5 voltios proporcionados por el ordenador.
- Para cargar un sketch al microcontrolador utilizando la interfaz USB-Serie.

Conector Jack: Es utilizado para alimentar la placa cuando no esté conectada al ordenador o se utilice una alimentación superior a los 5 voltios. La alimentación aplicada a este conector debe estar entre los 7 y 12 voltios. Esto se debe al regulador de voltaje empleado.

Regulador de voltaje: son los encargados de garantizar un voltaje adecuado al resto de los componentes de un sistema. El regulador LD1117S50 es el principal y encargado de garantizar

la alimentación del microcontrolador y los pines +5V presentes en los cabezales de la placa. Éste toma un voltaje desde el conector jack o el pin VIN y ofrece a su salida 5V.

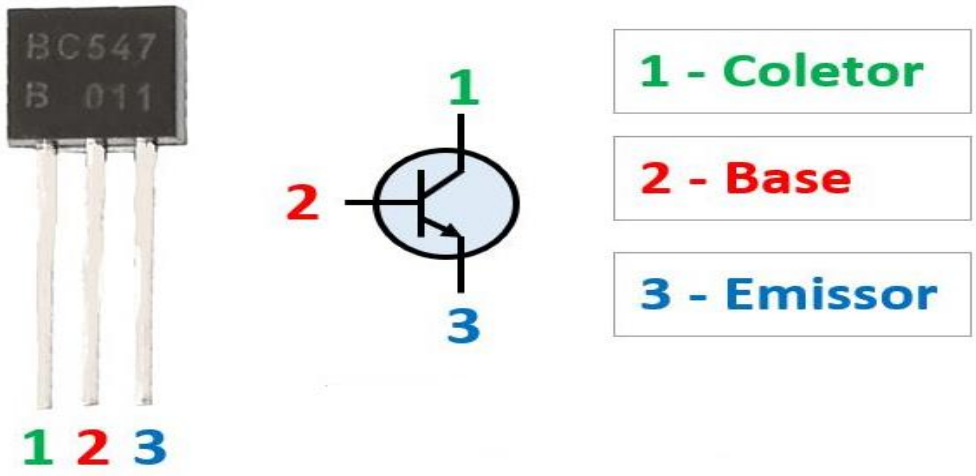


En la figura se observa que el conector jack se conecta al regulador mediante un diodo (esto es para proteger la placa si por error utilizas una fuente invertida). Los condensadores PC1, C2 y PC2 se encargan de estabilizar el voltaje a la entrada y salida del mismo.

Revisando la hoja de datos del regulador se puede comprobar que es capaz de ofrecer hasta 800 mA de corriente. Además, cuenta con protección ante cortocircuito y sobrecalentamiento.

Transistor BC547

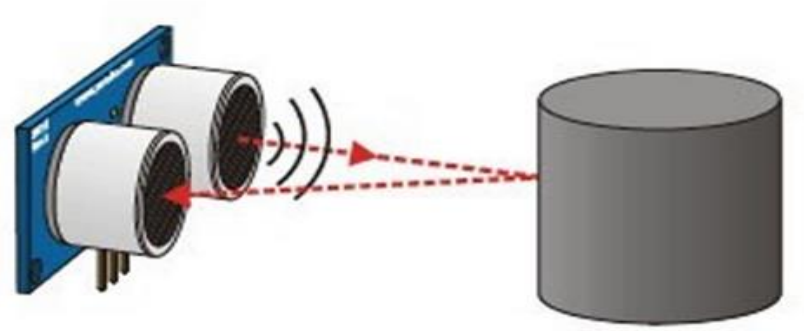
Es un transistor bipolar NPN. El dispositivo viene integrado en un encapsulado plástico tipo TO-92. El orden de los pines mirando la parte plana del encapsulado de derecha a izquierda es emisor, base y colector.



Sensor ultrasónico HC-SR04

El sensor HC-SR04 posee dos transductores: un emisor y un receptor piezoeléctricos, además de la electrónica necesaria para su operación. El funcionamiento del sensor es el siguiente: el emisor piezoeléctrico emite 8 pulsos de ultrasonido(40KHz) luego de recibir la orden en el pin TRIG, las ondas de sonido viajan en el aire y rebotan al encontrar un objeto, el sonido de rebote es detectado por el receptor piezoeléctrico, luego el pin ECHO cambia a Alto (5V) por un tiempo igual al que demoró la onda desde que fue emitida hasta que fue detectada, el tiempo del pulso ECO es medido por el microcontrolador y así se puede calcular la distancia al objeto. El funcionamiento del sensor no se ve afectado por la luz solar o material de color negro (aunque

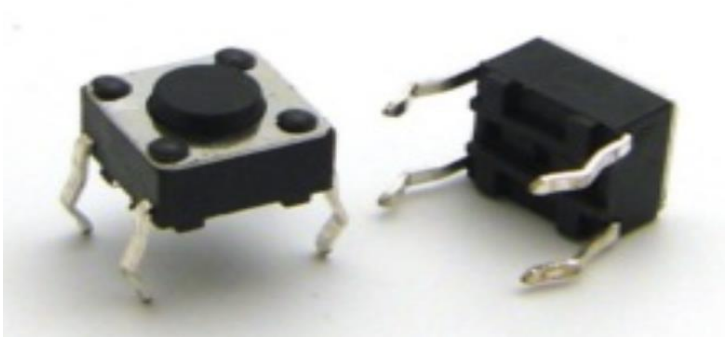
los materiales blandos acústicamente como tela o lana pueden llegar a ser difíciles de detectar).



Interruptor de botón (Pulsador)

Un pulsador o interruptor, es un dispositivo simple con dos posiciones, ON y OFF, un ejemplo es el interruptor de la luz. Estos pequeños pulsadores son de un 1/4 por cada lado, tienen 4 pastillas por lo que se puede pensar que hay 4 cables, pero son dos de cada lado unidos, por tanto, este pulsador es solamente de 2 cables.

El circuito estará abierto cuando el pulsador no esté presionado, de forma que al apretarlo, se cerrará el circuito, haciendo que pase la corriente a través de él.



Desarrollo

Para activar los leds en los semáforos, hemos desarrollado un programa en Arduino capaz de cambiar los estados de ambos semáforos se manera secuencial y ordenada. También hemos tenido en cuenta los estados de los pulsadores, los cuales simulan ser presionado por los peatones para cruzar la calle, y el de los sensores, los cuales simulan detectar el acercamiento de un vehículo.

```
#define TriggerSensor1 2
#define TriggerSensor2 3
#define EchoSensor1 4
#define EchoSensor2 5
#define Pulsador1 6
#define Pulsador2 7
#define LedRojo1 8
#define LedAmarillo1 9
#define LedVerde1 10
#define LedRojo2 11
#define LedAmarillo2 12
#define LedVerde2 13
int verde = 1;
int amarillo = 2;
int rojo = 3;
int estadoPulsador1 = 0;
int estadoPulsador2 = 0;
int estadoSensor1 = 0;
int estadoSensor2 = 0;
int semaforo1 = verde;
int semaforo2 = rojo;
int colorAnterior1 = 0;
int colorAnterior2 = 0;
long tiempoSensor1 = 0;
long distanciaSensor1 = 0;
long tiempoSensor2 = 0;
long distanciaSensor2 = 0;
void setup() {
  //Semaforo 1
  pinMode(LedRojo1, OUTPUT);
```

```
  pinMode(LedAmarillo1, OUTPUT);
  pinMode(LedVerde1, OUTPUT);
  //Semaforo 2
  pinMode(LedRojo2, OUTPUT);
  pinMode(LedAmarillo2, OUTPUT);
  pinMode(LedVerde2, OUTPUT);
  //Pulsador 1
  pinMode(Pulsador1, INPUT);
  //Pulsador 2
  pinMode(Pulsador2, INPUT);
  //Sensor 1
  pinMode(TriggerSensor1, OUTPUT);
  pinMode(EchoSensor1, INPUT);
  //Sensor 2
  pinMode(TriggerSensor2, OUTPUT);
  pinMode(EchoSensor2, INPUT);
  //Inicializacion
  digitalWrite(TriggerSensor1, LOW);
  digitalWrite(TriggerSensor2, LOW);
  prenderVerde(LedRojo1,LedAmarillo1,LedVerde1);
  prenderRojo(LedRojo2,LedAmarillo2,LedVerde2);
  delay(4000);
}
void loop() {
  //Iniciando secuencia de semaforo
  if (semaforo1 == verde){
    semaforo1 = amarillo;
    prenderAmarillo(LedRojo1,LedAmarillo1,LedVerde1);
    semaforo2 = amarillo;
    prenderAmarillo(LedRojo2,LedAmarillo2,LedVerde2);
```

```
  colorAnterior1 = verde;
  colorAnterior2 = rojo;
  delay(2000);
 }
 else if (semaforo1 == amarillo){
  if (colorAnterior1 == verde){
    semaforo1 = rojo;
    prenderRojo(LedRojo1,LedAmarillo1,LedVerde1);
    semaforo2 = verde;
    prenderVerde(LedRojo2,LedAmarillo2,LedVerde2);
  }
  else {
    semaforo1 = verde;
    prenderVerde(LedRojo1,LedAmarillo1,LedVerde1);
    semaforo2 = rojo;
    prenderRojo(LedRojo2,LedAmarillo2,LedVerde2);
  }
  delay(4000);
 }
 else if (semaforo1 == rojo){
  semaforo1 = amarillo;
  prenderAmarillo(LedRojo1,LedAmarillo1,LedVerde1);
  semaforo2 = amarillo;
  prenderAmarillo(LedRojo2,LedAmarillo2,LedVerde2);
  colorAnterior1 = rojo;
  colorAnterior2 = verde;
  delay(2000);
 }
 //Lectura de pulsadores
 delay (500);
```

```
if (digitalRead(Pulsador1) == HIGH){
  estadoPulsador1 = 1;
}
if (digitalRead(Pulsador2) == HIGH){
  estadoPulsador2 = 1;
}
//Lectura de sensor 1
digitalWrite(TriggerSensor1, HIGH);
delay(1);
digitalWrite(TriggerSensor1, LOW);
tiempoSensor1 = pulseIn (EchoSensor1, HIGH);
distanciaSensor1 = tiempoSensor1 / 59;
//Lectura de sensor 2
digitalWrite(TriggerSensor2, HIGH);
delay(1);
digitalWrite(TriggerSensor2, LOW);
tiempoSensor2 = pulseIn (EchoSensor2, HIGH);
distanciaSensor2 = tiempoSensor2 / 59;
//Repeticion la lectura de pulsadores y sensores
delay (500);
if (digitalRead(Pulsador1) == HIGH){
  estadoPulsador1 = 1;
}
if (digitalRead(Pulsador2) == HIGH){
  estadoPulsador2 = 1;
}
digitalWrite(TriggerSensor1, HIGH);
delay(1);
digitalWrite(TriggerSensor1, LOW);
tiempoSensor1 = pulseIn (EchoSensor1, HIGH);
```

```
distanciaSensor1 = tiempoSensor1 / 59;
digitalWrite(TriggerSensor2, HIGH);
delay(1);
digitalWrite(TriggerSensor2, LOW);
tiempoSensor2 = pulseIn (EchoSensor2, HIGH);
distanciaSensor2 = tiempoSensor2 / 59;
//Repeticion la lectura de pulsadores y sensores
delay (500);
if (digitalRead(Pulsador1) == HIGH){
  estadoPulsador1 = 1;
}
if (digitalRead(Pulsador2) == HIGH){
  estadoPulsador2 = 1;
}
digitalWrite(TriggerSensor1, HIGH);
delay(1);
digitalWrite(TriggerSensor1, LOW);
tiempoSensor1 = pulseIn (EchoSensor1, HIGH);
distanciaSensor1 = tiempoSensor1 / 59;
digitalWrite(TriggerSensor2, HIGH);
delay(1);
digitalWrite(TriggerSensor2, LOW);
tiempoSensor2 = pulseIn (EchoSensor2, HIGH);
distanciaSensor2 = tiempoSensor2 / 59;
//Repeticion la lectura de pulsadores y sensores
delay (500);
if (digitalRead(Pulsador1) == HIGH){
  estadoPulsador1 = 1;
}
if (digitalRead(Pulsador2) == HIGH){
```

```
  estadoPulsador2 = 1;
}
digitalWrite(TriggerSensor1, HIGH);
delay(1);
digitalWrite(TriggerSensor1, LOW);
tiempoSensor1 = pulseIn (EchoSensor1, HIGH);
distanciaSensor1 = tiempoSensor1 / 59;
digitalWrite(TriggerSensor2, HIGH);
delay(1);
digitalWrite(TriggerSensor2, LOW);
tiempoSensor2 = pulseIn (EchoSensor2, HIGH);
distanciaSensor2 = tiempoSensor2 / 59;
//Iniciando secuencia de pulsador 1
if (estadoPulsador1 == 1){
  semaforo1 = amarillo;
  prenderAmarillo(LedRojo1,LedAmarillo1,LedVerde1);
  semaforo2 = amarillo;
  prenderAmarillo(LedRojo2,LedAmarillo2,LedVerde2);
  colorAnterior1 = verde;
  colorAnterior2 = rojo;
  estadoPulsador1 = 0;
  delay(2000);
}
//Iniciando secuencia de pulsador 1
else if (estadoPulsador2 == 1){
  semaforo1 = amarillo;
  prenderAmarillo(LedRojo1,LedAmarillo1,LedVerde1);
  semaforo2 = amarillo;
  prenderAmarillo(LedRojo2,LedAmarillo2,LedVerde2);
  colorAnterior1 = rojo;
```

```
    colorAnterior2 = verde;
    estadoPulsador2 = 0;
    delay(2000);
  }
  //Iniciando secuencia de sensor 1
  else if (distanciaSensor1<5){
    semaforo1 = amarillo;
    prenderAmarillo(LedRojo1,LedAmarillo1,LedVerde1);
    semaforo2 = amarillo;
    prenderAmarillo(LedRojo2,LedAmarillo2,LedVerde2);
    colorAnterior1 = verde;
    colorAnterior2 = rojo;
    distanciaSensor1 = 0;
    delay(2000);
  }
  //Iniciando secuencia de pulsador 2
  else if (distanciaSensor2<5){
    semaforo1 = amarillo;
    prenderAmarillo(LedRojo1,LedAmarillo1,LedVerde1);
    semaforo2 = amarillo;
    prenderAmarillo(LedRojo2,LedAmarillo2,LedVerde2);
    colorAnterior1 = rojo;
    colorAnterior2 = verde;
    distanciaSensor2 = 0;
    delay(2000);
  }
}
void prenderVerde(int LedRojo, int LedAmarillo, int LedVerde){
  digitalWrite(LedRojo, LOW);
  digitalWrite(LedAmarillo, LOW);
```

```
  digitalWrite(LedVerde, HIGH);
}
void prenderAmarillo(int LedRojo, int LedAmarillo, int LedVerde){
  digitalWrite(LedRojo, LOW);
  digitalWrite(LedAmarillo, HIGH);
  digitalWrite(LedVerde, LOW);
}
void prenderRojo(int LedRojo, int LedAmarillo, int LedVerde){
  digitalWrite(LedRojo, HIGH);
  digitalWrite(LedAmarillo, LOW);
  digitalWrite(LedVerde, LOW);
}
```

Problemas y mejoras

Uno de los problemas con los que nos encontramos al desarrollar el proyecto fue la detección del momento en el que se presiona el pulsador, ya que la lectura la realizamos cada medio segundo. Si la persona presiona el botón entre el tiempo de espera entre una lectura y otra, el programa no detecta el cambio y seguiría funcionando como si no se hubiese presionado. Esto se podría mejorar realizando mas mediciones cada menos tiempo para reducir la probabilidad que se pierda la medición.

Otra de las mejoras que se podría realizar es una optimización en el cambio de los colores del semáforo cuando se detecta que se presiono el botón. Actualmente para tener una menor complejidad en el código, al momento de presionar el pulsador, cambiamos el color del semáforo a amarillo y luego al color que corresponde, ignorando el estado actual del semáforo en ese momento.

Conclusión

En el transcurso del desarrollo del proyecto nos hemos encontrado con diversos inconvenientes, los cuales fuimos solventando con la adquisición de nuevos componentes, ajustes en las mediciones y cálculos, depuración del código del programa, entre otros.

De acuerdo a los conocimientos adquiridos en la materia y al material de consulta, hemos podido desarrollar un proyecto utilizando distintos tipos de componentes, como por ejemplo los transistores en conjunto con los leds, pulsadores y sus configuraciones en pull-up/pull-down, los sensores de distancia ultrasónicos y aprendiendo de su uso en la práctica integrándolos con el Arduino, siendo un punto muy importante para el perfeccionamiento de los conocimientos teóricos vistos en la materia durante el cuatrimestre.