DISPOSITIVO AUTOMATIZADO DE SENSORES PARA ARDUINO (DASA)

MANUAL DE USUARIO

V2.4 PCB v9a

Miguel Ángel Bañuelos Saucedo, ICAT
Milagros Pacheco Castañeda, ENP-5
Rebeca Guillermina Villegas Salas, ENP-7

Proyecto PAPIME PE101620

Desarrollo de material didáctico para la asignatura Informática Aplicada a la Ciencia y a la Industria

Universidad Nacional Autónoma de México

Dirección General de Asuntos del Personal Académico

Proyecto PAPIME PE101620

Ciudad de México, 2021

Contenido

INTRODUCCIÓN 4
1. DESCRIPCIÓN 4
2. FUNCIONAMIENTO 6
2.1. Display LCD 7
2.2. Conexión de un potenciómetro (pin A0) 9
2.3. Generación de un tono (zumbador pasivo en el pin A1) 11
2.4. Fotorresistencia (pin A2) 12
2.5. Botón (pin A2) 13
2.6. Sensor de temperatura LM35 (pin A3) 15
2.7. Sensor de presión MPX5700 (pin A4) 17
2.8. Sensores de gas MQ-n (pin A5) 22
2.9. Indicadores LED (pines D3, D4 y D5) 25
2.10. Sensor de ultrasonido HC-SR04 (pines D6 y D7) 26
2.11. Servomotor (pin D6) 28
2.12. Motor de corriente directa (pin D3) 30
2.13. Sensor de humedad y temperatura DHT-11 (conexión externa) 32
2.14. Sensor de presencia (PIR) 36
2.15. Ventilador 39
ANEXO A. Programa básico de prueba 40
ANEXO B. Contenido del kit 42
ANEXO C. Diagrama esquemático 43
ANEXO D. SERIGRAFÍA 44
ANEXO E. Lista de partes del módulo DASA 45

INTRODUCCIÓN

Las tarjetas Arduino deben su popularidad a su facilidad de uso, bajo costo, y a la gran cantidad de programas de ejemplo disponibles. También cuenta con numerosas bibliotecas de libre acceso para el manejo de sensores y actuadores. Una tarjeta Arduino constituye una excelente herramienta de aprendizaje de diversos temas que van desde los principios de la informática, hasta conceptos de electrónica y manejo de sensores y actuadores. Esto posibilita la adopción de las tarjetas Arduino como apoyo a la impartición de diversas asignaturas de nivel preuniversitario y universitario.

Uno de los problemas que pueden surgir al utilizar las tarjetas Arduino es el proceso de conexión de los componentes. En muchos casos, los estudiantes cuentan con poca experiencia en el seguimiento de diagramas y manejo de componentes electrónicos. Ello puede resultar en una demora en el desarrollo de la práctica debido al tiempo empleado para armar y revisar aquellos circuitos que tienen algún problema de montaje.

Como una opción para reducir el tiempo necesario para realizar conexiones se desarrolló el Dispositivo Automatizado de Sensores para Arduino (DASA). Este dispositivo es un módulo escudo (*shield*) que se inserta sobre una placa Arduino UNO o similar compatible. Cuenta con un display LCD alfanumérico y conexión a diversos sensores y actuadores, algunos de los cuales ya están integrados en el módulo.

En este documento se presentan los componentes de los que está formado el DASA, el diagrama de conexiones y ejemplos de su operación, que incluyen el programa respectivo. El presente manual está orientado a personas familiarizadas con el manejo y programación de una tarjeta Arduino UNO, y con experiencia básica en manejo de componentes electrónicos.

1. DESCRIPCIÓN

Uno de los dispositivos más empleados en experimentos con Arduino y que requiere de varias conexiones es la pantalla LCD alfanumérica. Es por ello, que el módulo que se desarrolló cuenta con una pantalla de este tipo. Adicionalmente, el DASA cuenta con sensores, indicadores LED, un indicador acústico (zumbador pasivo) y conexiones directas para sensores externos, y motores (corriente directa y servomotor). Las conexiones también pueden utilizarse con otros dispositivos mediante una tarjeta protoboard pequeña, pero la mayor parte de las conexiones ya estarán incluidas en el módulo DASA (ver Figura 1).

Figura 1. El módulo DASA siendo montado en una placa Arduino UNO.

Los dispositivos con que cuenta el módulo DASA y que ya se encuentran montados en la placa se muestran en la Tabla 1. Existen además algunos conectores que están diseñados para montar de manera temporal componentes adicionales (sensores y actuadores) los cuales se indican en la Tabla 2.

Tabla 1. Dispositivos integrados en la placa del DASA.

Dispositivos
Display LCD alfanumérico de 16 x 2 caracteres
Zumbador pasivo
Fotorresistencia
Sensor de temperatura LM35
Interruptor óptico CNY70
LED rojo
LED amarillo
LED verde
Push-button

Tabla 2. Sensores y actuadores externos.

Sensores y actuadores	**Conector utilizado**
Potenciómetro	Bloque de terminales con tornillo
Sensor de presión MPX5700DP	Header hembra 6 pines
Sensor de gas de tipo MQ-2, MQ-3, o MQ-7	Header hembra 6 pines
Sensor ultrasónico HC-SR04.	Header hembra 4 pines
Motor de corriente directa	Bloque de terminales con tornillo
Servomotor de CD	Header macho 3 pines

El módulo Dispositivo Automatizado de Sensores para Arduino se basa en una pantalla LCD alfanumérica de 16x2 caracteres. El LCD utiliza seis líneas digitales del Arduino. El resto de las líneas se utilizan para los sensores, indicadores LED, zumbador y conexión de dispositivos externos. Se dejan libres las líneas D0 y D1 para ser utilizadas por la comunicación serial, si así lo requiere el usuario. De esta manera, siempre es posible enviar datos de los sensores a una computadora mediante la conexión USB. Los elementos principales del módulo se muestran en la Figura 2.

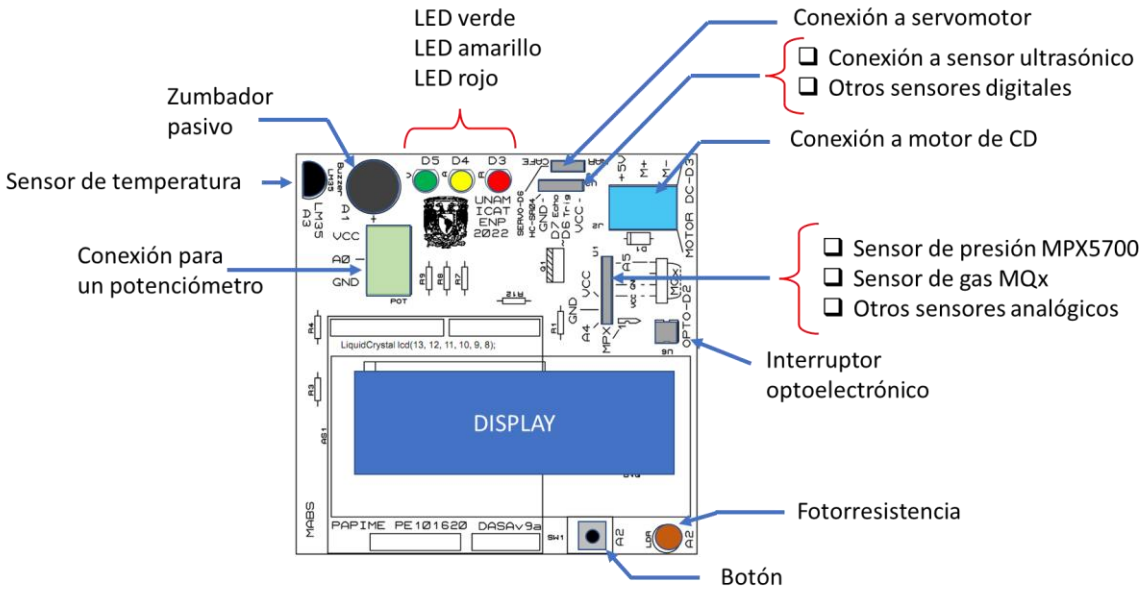


Figura 2. Sensores, actuadores y conexiones del DASA.

Para facilitar el uso del módulo, en la serigrafía se indica el pin analógico o digital que corresponde a cada uno de los dispositivos integrados. También se señala el tipo de dispositivo externo que se puede conectar en los *headers* y las líneas de alimentación y tierra, para utilizar otro tipo de sensores y actuadores externos.

2. FUNCIONAMIENTO

La operación del módulo DASA se basa en el control de líneas analógicas y digitales de la tarjeta Arduino. En la Tabla 3 se muestra cómo se utilizan las conexiones digitales con una tarjeta Arduino UNO. Como se mencionó anteriormente las líneas D0 y D1 quedan libres para utilizarse en la comunicación USB. Las conexiones analógicas se enlistan en la Tabla 4. La línea A1 se utiliza como salida digital para el zumbador pasivo.

Tabla 3. Conexiones digitales del módulo DASA.

Pin Arduino UNO	Conexión
D13	LCD-RS
D12	LCD-E
D11	LCD-D4
D10	LCD-D5
D9	LCD-D6
D8	LCD-D7
D7	Sensor de ultrasonido (Echo)
D6	Sensor de ultrasonido (Trigger)
D5	Led verde
D4	Led amarillo
D3	Led rojo/Motor C.D.
D2	Optointerruptor CNY70

Tabla 4. Conexiones analógicas del módulo DASA

Pin Arduino UNO	Conexión
A0	POT. Bloque de terminales con tornillo
A1	Salida **digital** para el Zumbador pasivo
A2	Fotorresistencia/Botón
A3	Sensor de temperatura LM35
A4	Sensor de presión MPX5700DP (U1)
A5	Sensor de gas MQ-x (U1)

2.1. Display LCD

El módulo cuenta con un display LCD alfanumérico de 16x2 caracteres. Para el manejo del display LCD se requieren seis líneas digitales (ver la Figura 3). Las líneas de control (RS, Register Select) y (E, Enable) del LCD se conectan a las líneas D13 y D12 de la tarjeta Arduino UNO, respectivamente. Por otro lado, las líneas de datos del display LCD (D4-D7) se conectan a los pines D11-D8 de la tarjeta Arduino. Esto último es importante recordarlo al realizar la configuración de la biblioteca de manejo del LCD en el programa. Las líneas de datos D0-D3 del LCD no se utilizan. En la parte inferior de la figura se encuentran las resistencias que fijan el nivel de contraste del display y sustituyen al potenciómetro que se utiliza en algunas ocasiones; es decir, el contraste no se podrá cambiar. *Por simplicidad, las líneas de alimentación y de ajuste de contraste se omiten en otras subsecciones de este documento.*

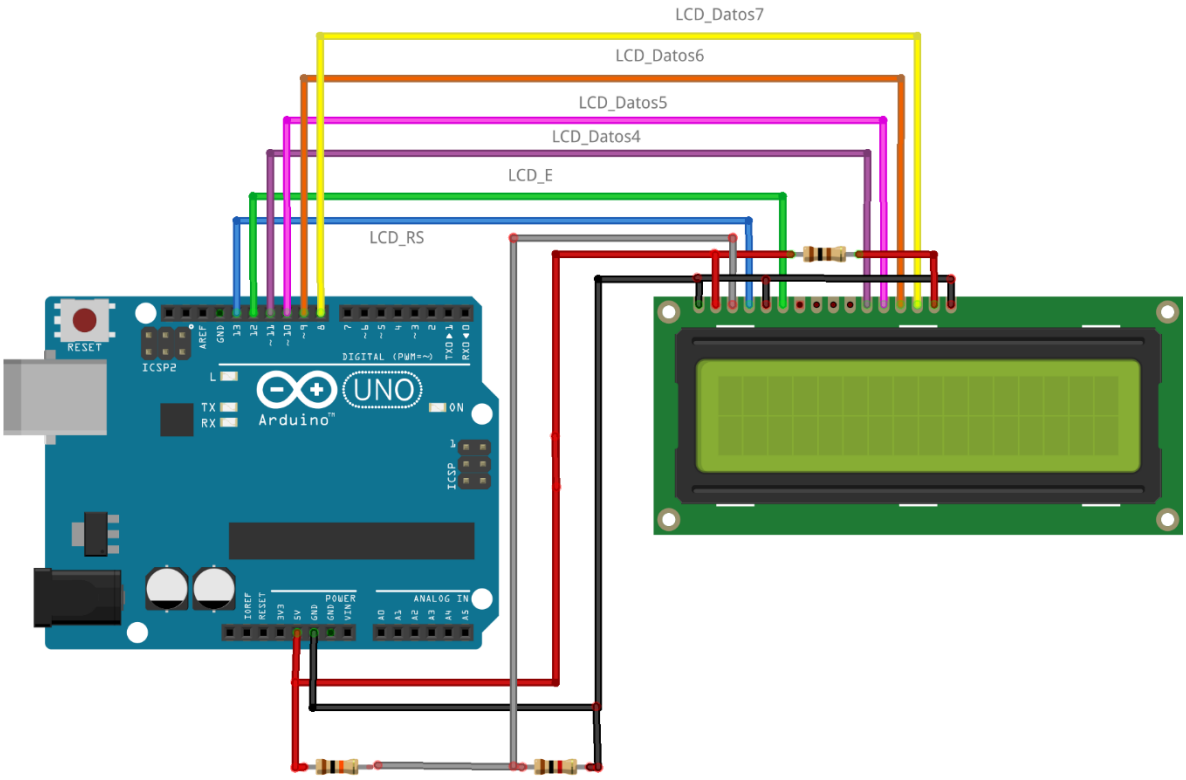


Figura 3. Diagrama de conexiones del display LCD.

Como ejemplo de la utilización del display, se muestra el código del programa LCD Hola mundo en la Tabla 5. Este programa configura los pines de comunicación con el LCD y envía los mensajes “Hola Mundo” y “Modulo DASA”. Es importante recordar que las vocales acentuadas no están definidas en el controlador del LCD, aunque se pueden generar por programa.

Tabla 5. Programa Hola mundo para el LCD.

```
/*  LCD_Hola_mundo
 *  Programa que despliega un mensaje en un display LCD
    UNAM 2021*/
// Se requiere la biblioteca de uso del display
#include <LiquidCrystal.h>

// Se definen los pines que se comunican con el display
//            lcd(RS, E, D4, D5, D6, D7);
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);

// Inicialización
void setup() {
   lcd.begin(16, 2);   // LCD es de 2 renglones y 16 caracteres
   lcd.print("   Hola Mundo"); // Manda un letrero al display
   //lcd.setCursor(col, row)
   lcd.setCursor(0,1);  //Selecciona segundo renglón
   lcd.print("  Modulo DASA");
}
// Programa principal
void loop() {
}
```

2.2. Conexión de un potenciómetro (pin A0)

En el bloque de terminales con tornillo etiquetado POT (ver la Figura 4) se puede atornillar un potenciómetro y realizar alguna práctica utilizando el valor de voltaje que entrega al pin analógico A0. Por ejemplo, se puede utilizar para controlar la velocidad de un motor de corriente directa (CD), o la intensidad de un led.

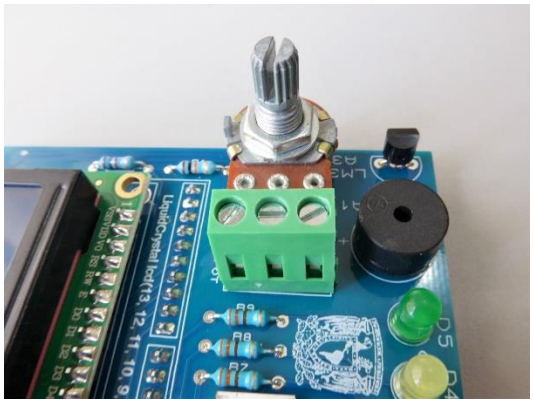


Figura 4. El potenciómetro conectado a la terminal con tornillos POT.

Para una correcta conexión del potenciómetro, es importante identificar cuándo la posición del borne está abierta o cerrada. Si no se introducen los pines del potenciómetro en posiciones abiertas, entonces no quedará bien sujetado. Para abrir una posición es necesario girar en sentido antihorario el tornillo correspondiente. Un detalle de cómo se puede identificar visualmente el estado de cada posición del borne se presenta en la Figura 5.

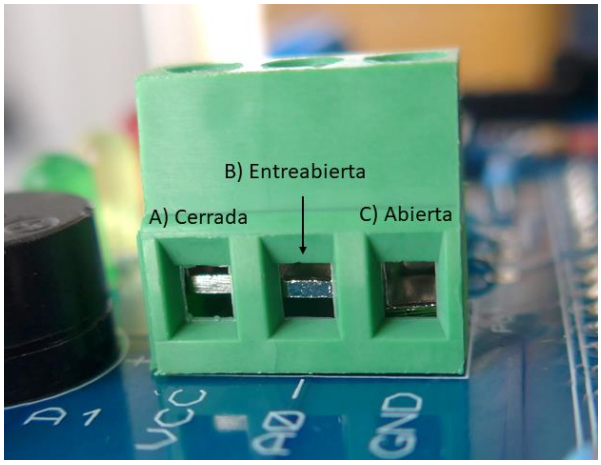


Figura 5. Identificación del estado de las posiciones del borne. A) Cerrada, B) entreabierta, y C) abierta.

Como ejemplo del manejo del potenciómetro se muestra el código de un programa que lee el valor de voltaje que entrega el potenciómetro al pin A0 y el valor que produce la conversión analógico-digital (es decir, un valor entre 0 y 1023) se muestra en el display LCD (ver Tabla 6). Un ejemplo de la ejecución de este programa se muestra en la Figura 6.

Tabla 6. Programa que muestra en el LCD el valor del convertidor A/D del pin A0.

```
/*  LCD_POT
  *  Programa que lee un voltaje generado por un potenciómetro
  *  y muestra el valor del convertidor analógico-digital
  *  en el display LCD
  *  NOTA: El potenciómetro se conecta al pin A0
  *  UNAM 2021
 */

// include the library code:
#include <LiquidCrystal.h>

// Inicializa las conexiones del LCD
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
const int pinPot = 0;   // Potenciómetro en el pin A0
int valorPotenciometro = 0;  //Resultado del convertidor A/D

void setup() {
  pinMode(pinPot, INPUT);
  lcd.begin(16, 2);
  // Mensaje en el primer renglón
  lcd.print("El valor es:");
}

void loop() {
  lcd.setCursor(0, 1);  //Selecciona segundo renglón
  valorPotenciometro = analogRead(pinPot);
  lcd.print("      ");   //Borra el valor anterior
  lcd.setCursor(0, 1);
  lcd.print(valorPotenciometro);
  delay(100);  //Retardo de 100 ms
}
```

Figura 6. Ejemplo del mensaje que produce el programa LCD POT.

El circuito que se requiere para hacer funcionar el programa LCD POT de la Tabla 6 se muestra en la Figura 7. Se han omitido las resistencias de ajuste del nivel de contraste y la resistencia de control de intensidad de la retroiluminación.

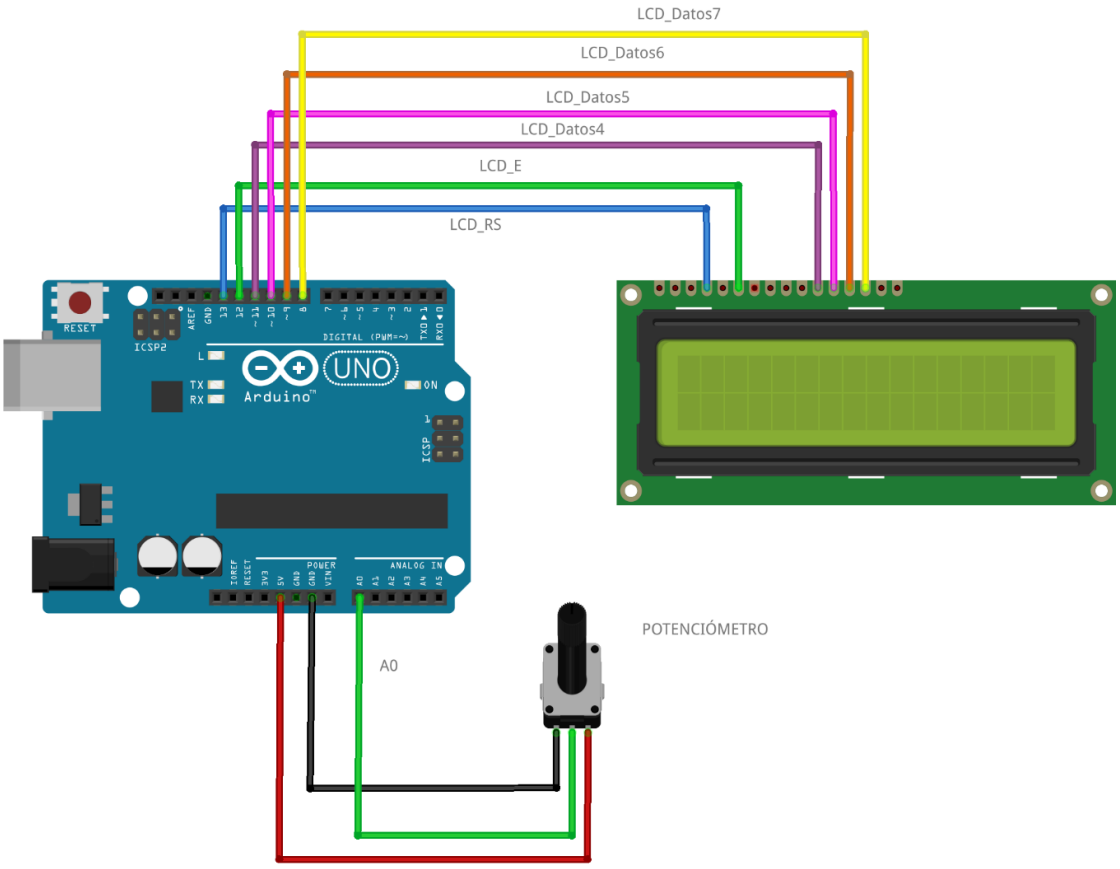


Figura 7. Circuito para el ejemplo LCD POT.

2.3. Generación de un tono (zumbador pasivo en el pin A1)

Los pines analógicos también se pueden utilizar como salidas digitales, para hacerlo, basta con configurar el pin A0 a A5 como si fuera un pin digital (p. ej: pinMode(A1, OUTPUT)😉. Un zumbador pasivo se encuentra conectado al pin A1 y se encuentra en una esquina de la tarjeta (ver Figura 8).

Figura 8. Zumbador pasivo.

Los zumbadores pasivos pueden generar diferentes tonos, mientras que los zumbadores activos emiten un solo tono. El lenguaje de Arduino permite generar un tono de frecuencia específica mediante la instrucción *tone(pin, frec);*. Como ejemplo, se presenta el código de la Tabla 7.

Tabla 7. Programa Zumbador.

```
/* Zumbador
 * Programa que produce un tono mediante un
 * zumbador pasivo conectado al pin A1
 * UNAM 2021
 */
void setup() {
  // Se configura el pin A1 como salida digital
  pinMode(A1, OUTPUT);
  // Se genera un tono de 440 Hz en el pin A1
  // El tono dura 2 segundos y se apaga
  tone(A1,440);
  delay(2000);
  noTone(A1);  // Se apaga el zumbador
}

void loop() {
}
```

2.4. Fotorresistencia (pin A2)

El módulo incluye una fotorresistencia de 0.5 MΩ conectada en serie con una resistencia de 10 kΩ (ver Figura 9). Las siglas LDR son el acrónimo de *Light Dependant Resistor* (Resistor dependiente de la luz). El valor de 0.5 MΩ representa la resistencia en ausencia de luz, aunque puede haber variaciones de dispositivo a dispositivo, y no todas las fotorresistencias tienen este valor. El punto intermedio entre la fotorresistencia y la resistencia se encuentra conectado al pin A2, el cual, como ya se comentó, puede configurarse como entrada analógica o digital. En el caso de que se configure como entrada digital, el circuito generará un nivel ALTO de voltaje cuando no hay suficiente iluminación, y un nivel BAJO cuando sube el nivel de iluminación. Este procedimiento se ejemplifica en el programa de la Tabla 8.

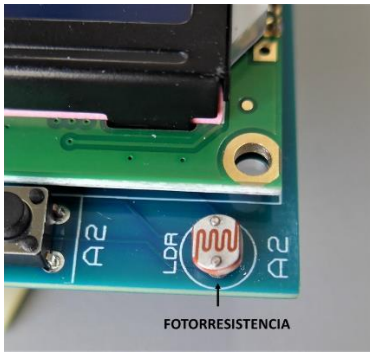


Figura 9. Fotorresistencia.

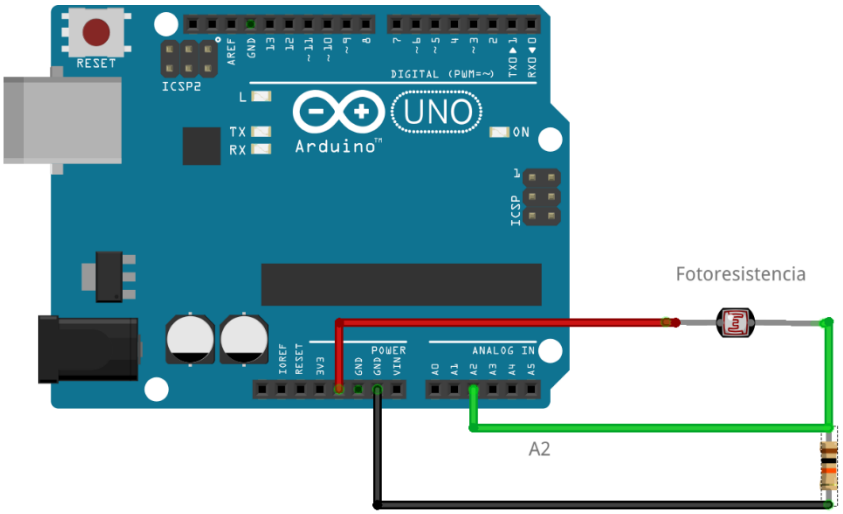


Figura 10. Diagrama de conexiones de la fotorresistencia.

Tabla 8. Programa que muestra el funcionamiento del circuito con fotorresistencia.

```
/* LED_foto_res
 * Programa que enciende un LED dependiendo de la
 * iluminación que recibe la fotorresistencia
 * UNAM 2021
 */

  // pin de la fotorresistencia
const int LED=3; // pin del LED rojo
const int fotor=A2; // Fotorresistencia en A2

void setup() {
   pinMode(fotor, INPUT); //A2 como entrada digital
   pinMode(LED, OUTPUT);  //LED rojo
}

void loop() {
  while(!digitalRead(fotor)){
    digitalWrite(LED, HIGH); //LED enciende
  }
  digitalWrite(LED, LOW); // LED se apaga
}
```

2.5. Botón (pin A2)

Durante la etapa final del diseño de la tarjeta DASA, se evaluó la posibilidad de incluir un botón (*push-button*). Debido a que ya se encontraban asignados todos los pines, se optó por conectar un botón compartiendo el pin A2 de la tarjeta Arduino. El botón está conectado en paralelo con la resistencia de 10 kΩ asociada a la fotorresistencia (ver Figura 11). De esta manera, si se configura el pin A2 como una entrada digital, se leerá un valor bajo “0” cuando el botón esté presionado, y un valor alto “1” si el botón está liberado. Es importante mencionar la fotorresistencia debe estar

iluminada para que el funcionamiento sea como el descrito. Un programa de ejemplo que enciende el led rojo al presionar el botón, se muestra en la Tabla 9. El botón se ubica junto a la fotorresistencia, en un costado de la tarjeta DASA, según se muestra en la Figura 12.

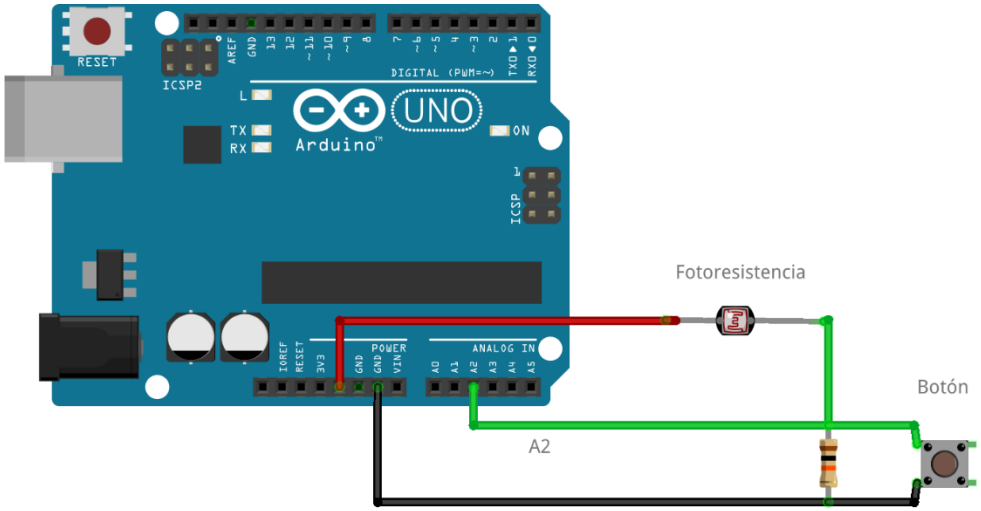


Figura 11. Diagrama de conexión del botón y la fotorresistencia.

Tabla 9. Código ejemplo del funcionamiento del botón.

```
/* LED_boton
 * Programa que enciende un LED dependiendo del
 * botón
 * UNAM 2021
 */

const int boton=A2; // A2 es el pin del boton
const int LED=3; // pin del LED rojo  D3

void setup() {
   pinMode(boton, INPUT); //A2 como entrada digital
   pinMode(LED, OUTPUT);  //LED rojo
}

void loop() {
  while(!digitalRead(boton)){
    digitalWrite(LED, HIGH); //LED enciende
  }
  digitalWrite(LED, LOW); // LED se apaga
}
```

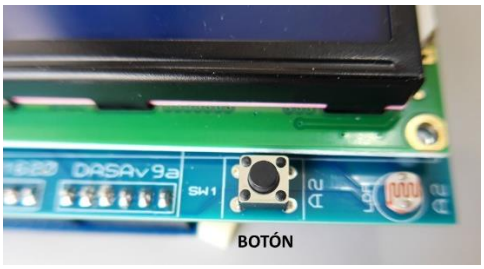


Figura 12. Fotografía del área de la placa donde se encuentra al botón.

2.6. Sensor de temperatura LM35 (pin A3)

El módulo DASA cuenta con un sensor de temperatura LM35 soldado a la placa y conectado al pin A3. Este sensor se puede utilizar para determinar la temperatura ambiente entre 0°C y 100 °C. En la Tabla 10, se muestra el diagrama de conexiones del sensor y del display LCD.

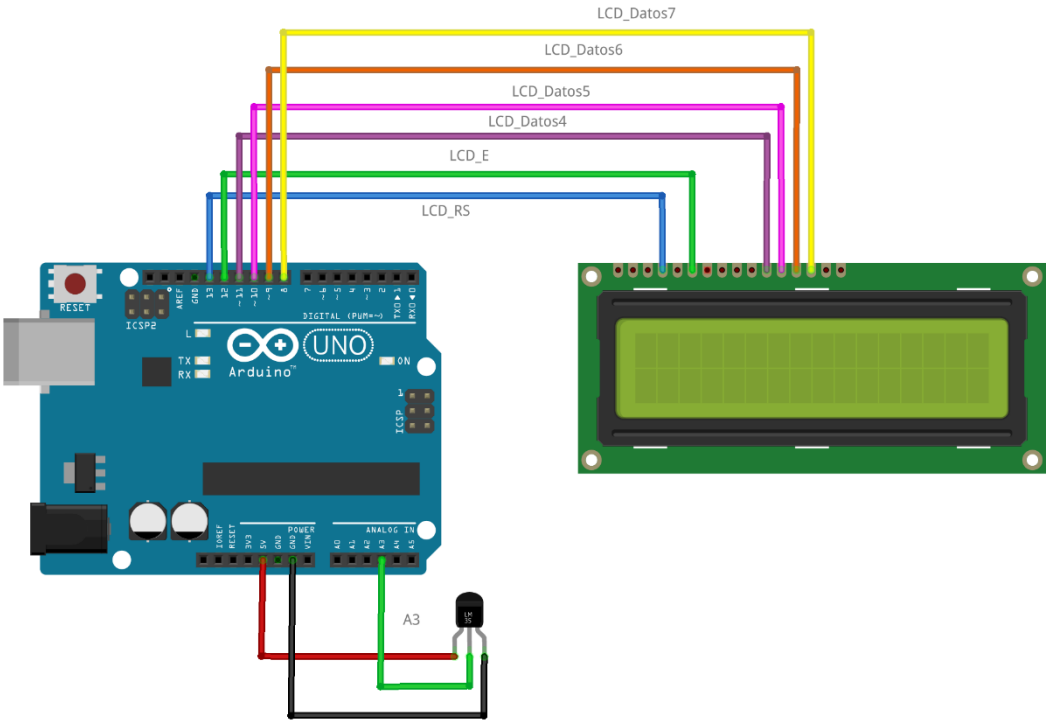


Figura 13. Conexión del sensor LM35 y el display LCD.

En la Tabla 10 se lista el código de un programa que utiliza la pantalla LCD para desplegar la temperatura que detecta el sensor LM35. En la Figura 14, se muestra un ejemplo de la operación de este programa.

Tabla 10. Programa que despliega la temperatura ambiente en el display LCD.

```
/*  LCD_LM35
 *  Programa que lee un voltaje generado por sensor LM35
 *  lo convierte al valor de temperatura ambiente en grados
 *  Celsius y lo muestra en el display LCD
 *  NOTA: El sensor se encuentra en el pin A3
 *  UNAM 2021
 */

#include <LiquidCrystal.h>
#define LM35 3  //Sensor de temperatura en pin A3

// Inicializa las conexiones del LCD
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
int valorSensor = 0;   //Almacena el dato del convertidor A/D
float temp = 0;   //Almacena el valor de la temperatura Celsius

void setup() {
  lcd.begin(16, 2);
  // Mensaje en el primer renglón
  lcd.print("  Sensor LM35");
}

void loop() {
valorSensor = analogRead(LM35);
temp = 500*(valorSensor/1023.0);
lcd.setCursor(0, 1);    //Segundo renglón
lcd.print("               "); //Borra el anterior
lcd.setCursor(0, 1);
lcd.print("Temp= ");
lcd.print(temp);
lcd.print(" ");
lcd.print(char(223)); //Símbolo de grado
lcd.print("C");
delay(500);  //Retardo de 500 ms
}
```

Figura 14. LCD mostrando el valor de la temperatura detectado con el sensor LM35.

El sensor de temperatura LM35 se ubica en una de las esquinas de la tarjeta, pues se encontró que en ese lugar está menos influenciado por el calor que genera la operación de la tarjeta Arduino y el display LCD (ver Figura 15).

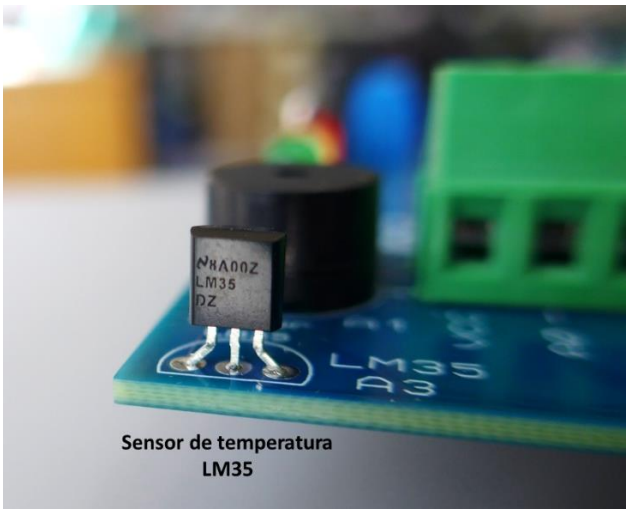


Figura 15. El sensor de temperatura LM35.

2.7. Sensor de presión MPX5700 (pin A4)

El módulo DASA cuenta con un conector *header* hembra de seis posiciones donde se puede introducir un sensor de presión MPX5700DP. Este sensor entrega un voltaje que es aproximadamente 6.4 mV por cada kPa, y puede detectar presiones de hasta 700 kPa; es decir, el equivalente a siete atmósferas. La señal del sensor está conectada al pin A4 de la tarjeta Arduino UNO. Para conectar correctamente al sensor es importante ubicar el pin número 1 del mismo, el cual tiene una pequeña muesca. La muesca también se representa en la serigrafía del módulo DASA tal como se muestra en la Figura 17.

El sensor cuenta con dos boquillas donde se puede acoplar una manguera de plástico de 4 mm de diámetro externo. La boquilla que se encuentra del lado de la identificación del sensor es la que se encarga de detectar presión positiva, por lo que se recomienda utilizar esta. Un ejemplo de la operación del sensor de presión y el display LCD se presenta en la Figura 18. El código de este ejemplo se enlista en la Tabla 11. Es importante mencionar que el sensor puede no entregar cero volts cuando hay una presión nula. Por lo tanto, se debe modificar la línea

```
const float ajusteCero = 29.78;
```

Inicialmente se debe hacer ajusteCero=0, luego cargar el programa y anotar el valor que aparezca en el display cuando no se aplica ninguna presión (salvo la atmosférica). Posteriormente se deberá cambiar el valor de la constante ajusteCero y hacerlo igual al valor anotado. De esa manera se habrá calibrado a cero al sensor.

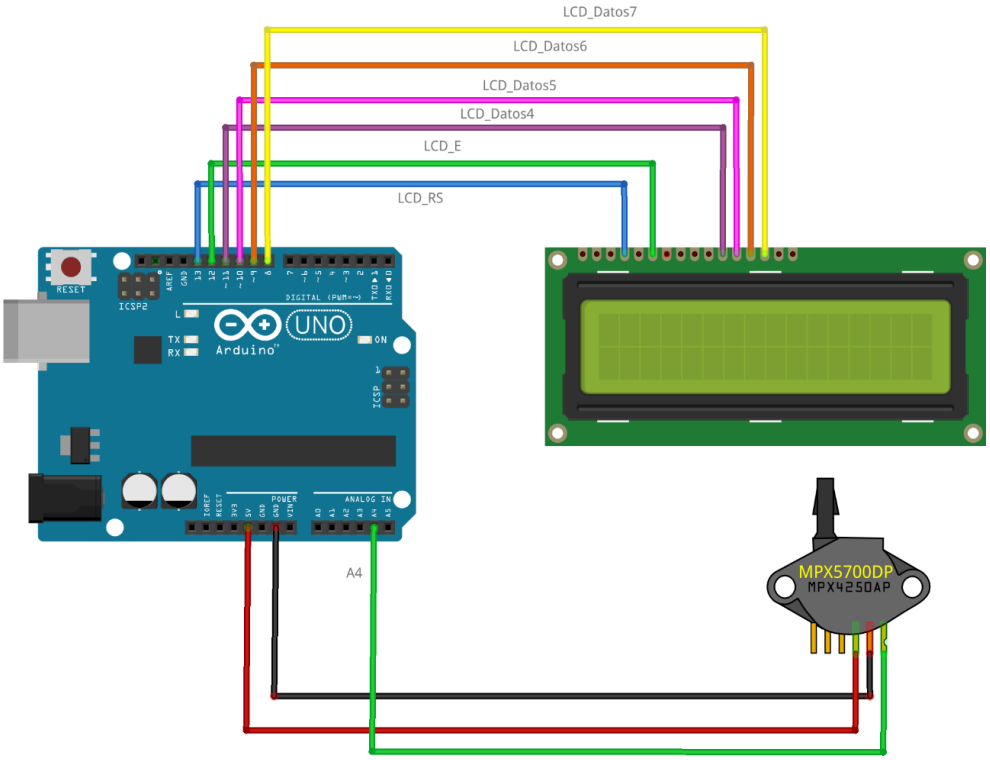


Figura 16. Diagrama de conexión del sensor MPX5700 y el display LCD.

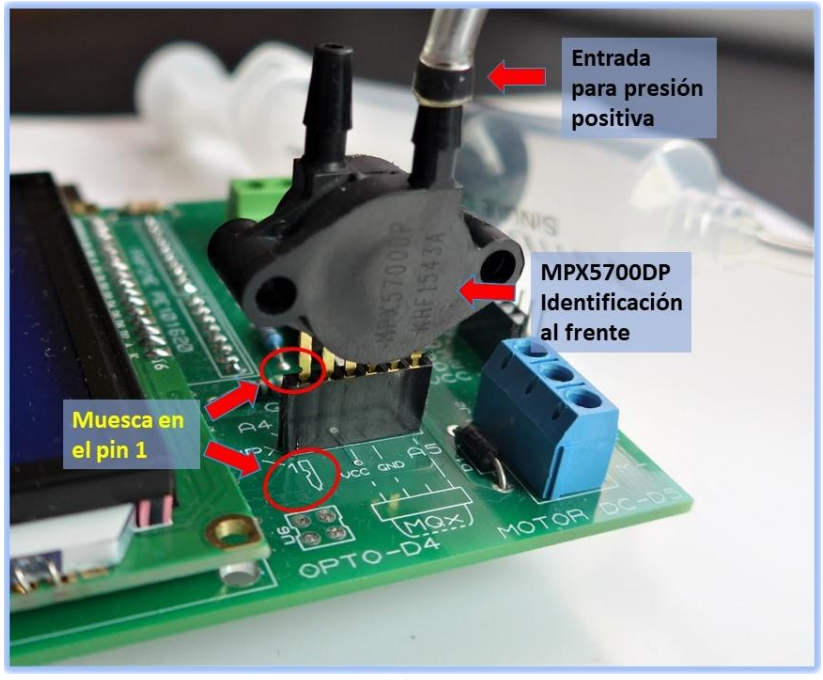


Figura 17. Montaje del sensor MPX5700.

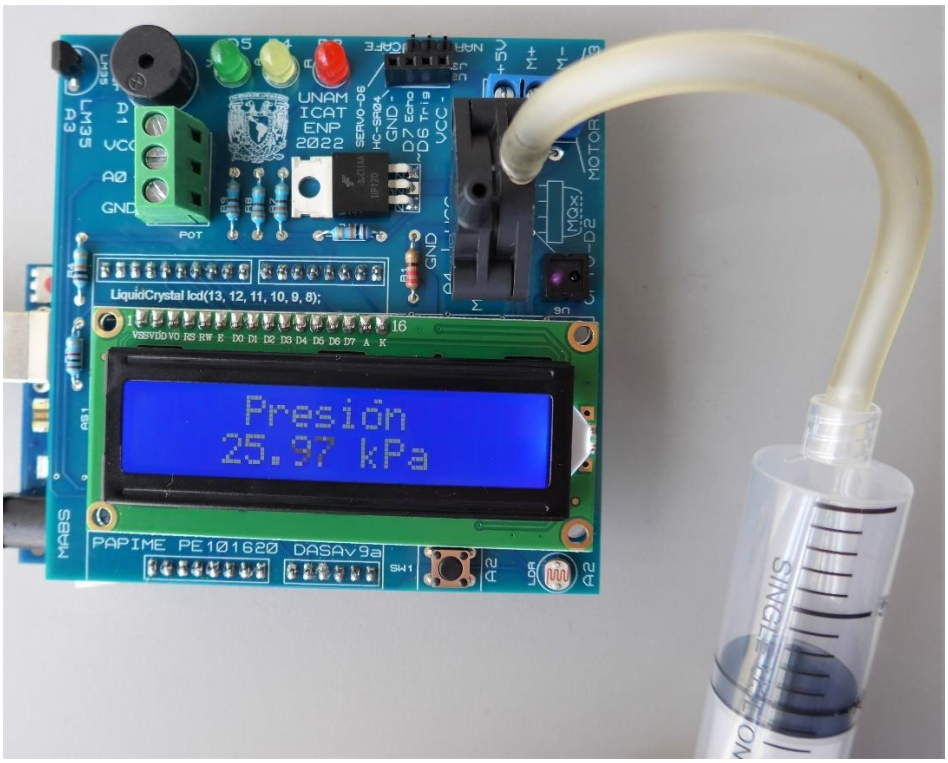


Figura 18. Ejemplo de operación del sensor de presión.

Tabla 11. Programa de ejemplo para el sensor MPX5700DP

```
/*  LCD_MPX5700
 *  Programa que lee el sensor de presión
 *  despliega el valor en un display LCD
    Se utiliza un sensor MPX5700DP
  UNAM 2021
*/
        #include <LiquidCrystal.h>
// Se definen los pines que se comunican con el display
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
byte o_acento[8] = {
  0b00011,
  0b00000,
  0b01110,
  0b10001,
  0b10001,
  0b10001,
  0b01110,
  0b00000
};  //Se crea una letra o acentuada
const int MPX = 4;   // A4 es el pin del sensor analógico
const float ajusteCero = 29.78;   // Hace el ajuste a cero en kPa
int valorSensor = 0;   // Guarda el valor del convertidor A/D
float presion;   //Guarda el valor de la presión en kPa
float voltaje;   //Guarda el valor del voltaje calculado en mV

void setup() {
  lcd.createChar(0, o_acento);   // Crea el caracter en la posición 0
  lcd.begin(16, 2);   // LCD es de 16 caracteres y 2 renglones
  lcd.print("    Presi");
  lcd.write((byte)0);  // Manda una "ó"
  lcd.print("n");
}
// Programa principal
void loop() {
  valorSensor = analogRead(MPX);
  lcd.setCursor(0, 1);  // Selecciona caracter 0, fila 1 (segunda)
  lcd.print("                ");   // borra 16 caracteres
  lcd.setCursor(0,1);
  voltaje = 5000*(valorSensor/1023.0);  // Convierte a mV
  presion = voltaje/6.4 - ajusteCero;  // Convierte a kPa
  lcd.setCursor(3, 1);
  lcd.print(presion);
  lcd.print(" kPa");
delay(500);
}
```

2.8. Optointerruptor CNY70 (pin D2)

La tarjeta cuenta con un interruptor óptico de tipo reflexivo interconstruido del tipo CNY70. El sensor se encuentra en uno de los costados de la placa, según se puede ver en la Figura 19. Este optointerruptor consta de un LED infrarrojo (que aparece iluminado de un color morado tenue en la fotografía) y un fototransistor. El fototransistor actúa como un interruptor: abierto cuando no hay un haz reflejado y cerrado en la presencia de un haz reflejado. De esta manera, el pin D2 entrega un nivel LOW, cuando no hay reflexión del haz, y un nivel HIGH, cuando algún objeto refleja el haz infrarrojo. El diagrama de conexión se muestra en la Figura 20.

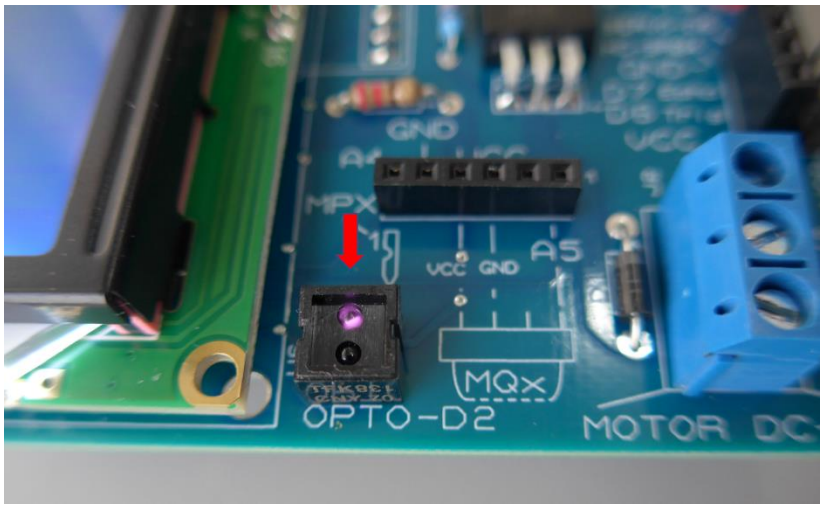


Figura 19. Ubicación del optointerruptor CNY70.

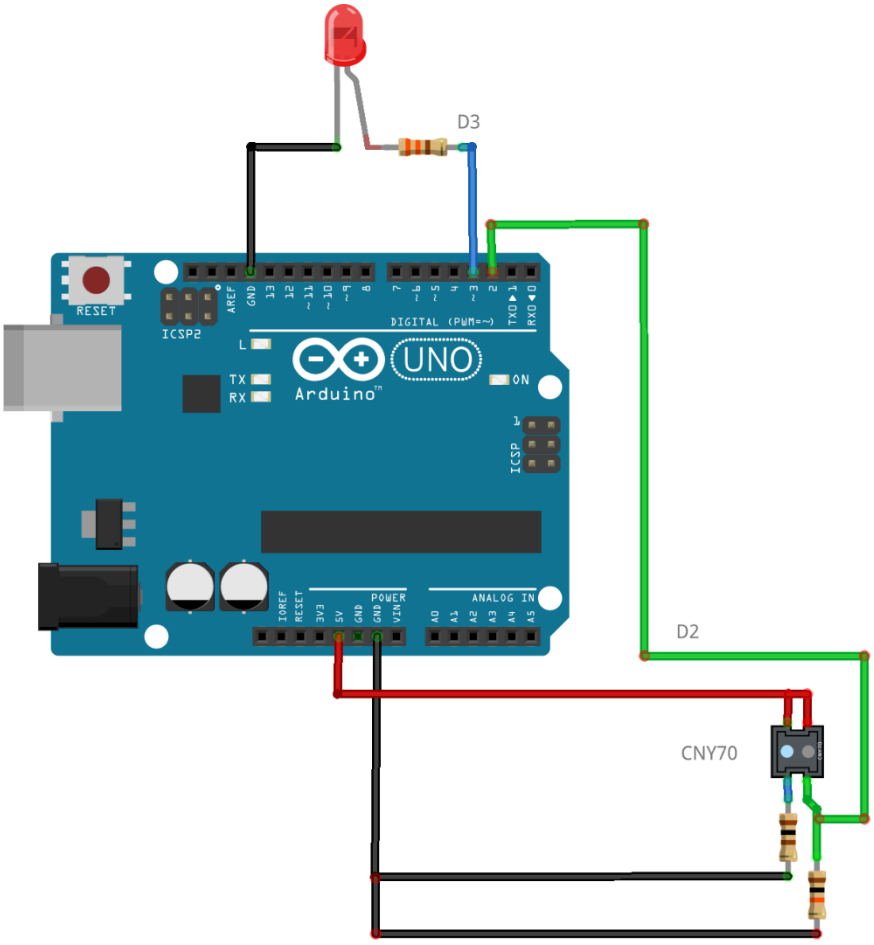


Figura 20. Diagrama de conexiones del optointerruptor CNY70 más un LED.

En la Tabla 12, se muestra el código de un programa que sirve para verificar el funcionamiento del optointerruptor. El programa enciende el LED rojo cada vez que se detecta un objeto frente al optointerruptor.

Tabla 12. Programa que prueba el funcionamiento del optointerruptor CNY70.

```
/*  DASA9_opto
  *  Programa que prueba el optointerruptor
  *  CNY70 encendiendo el LED rojo cuando hay
  *  un objeto que refleje el haz infrarrojo
     UNAM 2022*/

#define LEDR 3  //LED Rojo en el pin D3
#define opto 2  //Optointerruptor en pin D2


// Inicialización
void setup() {
  pinMode(LEDR, OUTPUT);
  pinMode(opto,INPUT);  //cny70
}
// Programa principal
void loop() {
  if(digitalRead(opto))
    digitalWrite(LEDR, HIGH);
  else
    digitalWrite(LEDR, LOW);
}
```

2.9. Sensores de gas MQ-n (pin A5)

El módulo DASA cuenta con un conector donde se puede insertar un sensor de gas del tipo MQ-x. Por ejemplo, puede utilizar un sensor de gas y humo MQ-2, un sensor de alcohol y gas MQ-3, o un sensor de monóxido de carbono MQ-7. Para ello se utiliza un *header* hembra de seis posiciones, utilizándose cuatro posiciones del conector, según se muestra en la Figura 21. Se utilizan las cuatro posiciones de un lado del conector, lo cual se indica en la serigrafía. La señal analógica del sensor está conectada al pin A5 de la tarjeta Arduino (ver Figura 22). La serigrafía también indica las conexiones VCC, GND y A5, que podrían utilizarse para conectar otro dispositivo. La posición entre GND y A5 no está conectada a ninguna señal.

Figura 21. Inserción de un sensor de gas MQ-x.

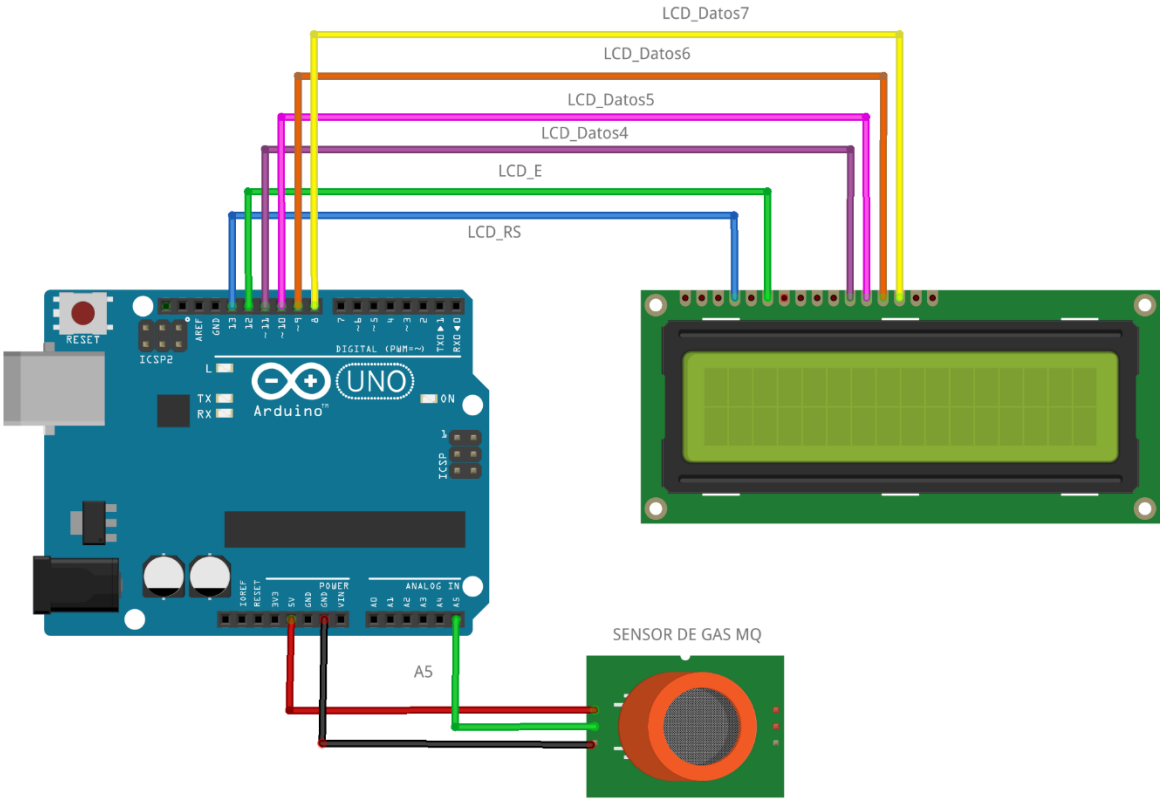


Figura 22. Diagrama de conexiones del sensor de gas MQ-x y el display LCD.

En módulo DASA, en operación con un sensor de gas MQ-2, se presenta en la Figura 23Tabla 13. Programa para el manejo del sensor MQ-2 y la pantalla LCD.. El código utilizado se puede consultar en la Tabla 13.

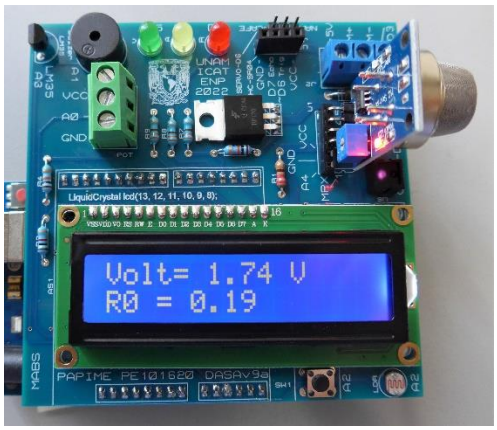


Figura 23. Ejemplo de operación del sensor de gas MQ-2 y el display LCD.

Tabla 13. Programa para el manejo del sensor MQ-2 y la pantalla LCD.

```
/* LCD_MQ2
 * Programa que lee el sensor de gas MQ-2
 * y despliega el valor en el LCD
  * Programa basado en
 * https://docs.particle.io/assets/datasheets/electronsensorkit/MQ-2.pdf
                                  * UNAM 2021
 */
#include <LiquidCrystal.h>
// Se definen los pines que se comunican con el display
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
float sensor_volt;  //Voltaje de salida del sensor
float RS_air; // Obtiene el valor de RS en aire limpio
float R0; // Obtiene el valor de R0 via H2
float sensorValue; //Acumula la lectura del convertidor A/D

void setup() {
lcd.begin(16, 2);   // LCD es de 2 renglones y 16 caracteres
}
void loop() {
/*--- Calcula el promedio de 100 lecturas ---*/
  for(int x = 0 ; x < 100 ; x++)
  {
    sensorValue = sensorValue + analogRead(A5);
  }
  sensorValue = sensorValue/100.0;  //Calcula el promedio
/*-----------------------------------------------*/
  sensor_volt = sensorValue/1024*5.0;  //Convierte a voltaje
  RS_air = (5.0-sensor_volt)/sensor_volt; // omit *RL
  R0 = RS_air/10.0; // La relación RS/R0 es 10 en aire puro
  lcd.setCursor(0,0);  //Selecciona primer renglón
  lcd.print("                ");  //Borra el anterior
  lcd.setCursor(0,0);  //Selecciona primer renglón
  lcd.print("Volt= "); // Manda un letrero al display
  lcd.print(sensor_volt);
  lcd.print(" V");
  lcd.setCursor(0,1);  //Selecciona segundo renglón
  lcd.print("                ");  //Borra el anterior
  lcd.setCursor(0,1);  //Selecciona segundo renglón
  lcd.print("R0 = ");
  lcd.print(R0);
  delay(1000);
}
```

2.10. Indicadores LED (pines D3, D4 y D5)

El módulo DASA cuenta con tres indicadores LED, que pueden utilizarse para programar un semáforo o alguna alarma luminosa. Hay un LED rojo, uno amarillo y uno verde, conectados a las líneas digitales D3, D4 y D5, respectivamente. Esto también se indica en la serigrafía de la placa electrónica. Los LED se conectan mediante una resistencia de limitación de corriente, tal como se indica en la Figura 24. Un programa que enciende en secuencia los tres LED se muestra en la Tabla 14.

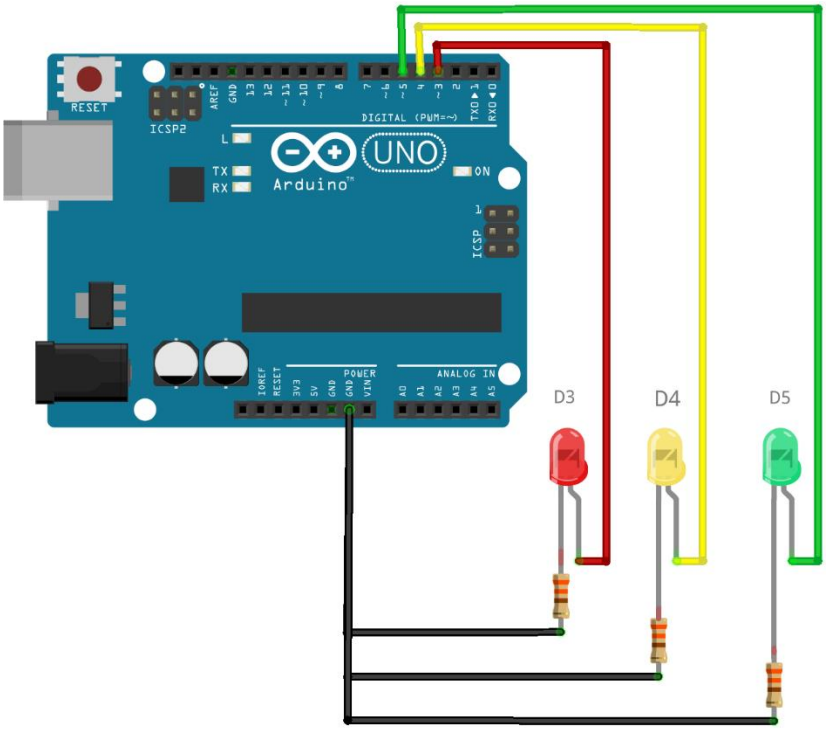


Figura 24. Diagrama de conexión de los indicadores LED.

Tabla 14. Programa que enciende los indicadores LED en secuencia.

```
/*   3LED
 *   Programa que prueba los tres leds
 *   encendiéndolos en secuencia
     UNAM 2021*/

#define LEDR 3   //LED Rojo en el pin D3
#define LEDA 4   //LED Amarilo en el pin D4
#define LEDV 5   //LED Verde en el pin D5

void setup() {
  pinMode(LEDR, OUTPUT);
  pinMode(LEDA, OUTPUT);
  pinMode(LEDV, OUTPUT);
}
void loop() {
  digitalWrite(LEDR, HIGH);
  delay(500);
  digitalWrite(LEDR, LOW);

  digitalWrite(LEDA, HIGH);
  delay(500);
  digitalWrite(LEDA, LOW);

  digitalWrite(LEDV, HIGH);
  delay(500);
  digitalWrite(LEDV, LOW);
}
```

2.11. Sensor de ultrasonido HC-SR04 (pines D6 y D7)

Al módulo DASA se le puede conectar un sensor de ultrasonido HC-SR04 en un conector header hembra de cuatro terminales (ver Figura 25). Con este sensor se pueden realizar experimentos para determinar la distancia a un objeto. Las terminales Trig y Echo del sensor están conectadas a las líneas digitales D6 y D7, respectivamente. Esto se indica en el diagrama de conexiones de la Figura 26.

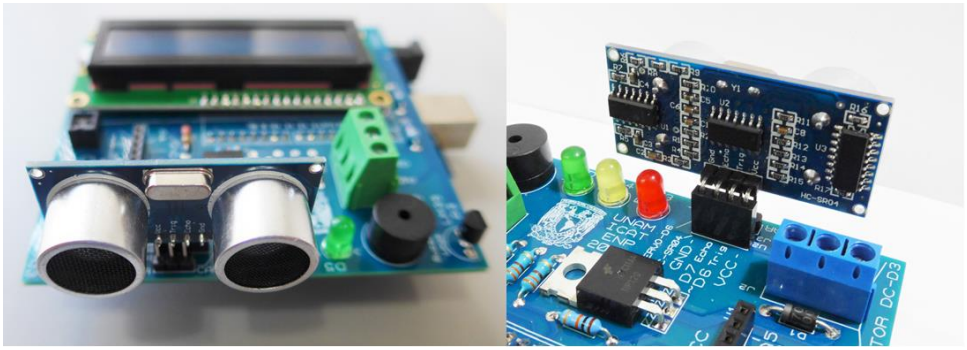


Figura 25. El sensor de ultrasonido montado en el módulo DASA.

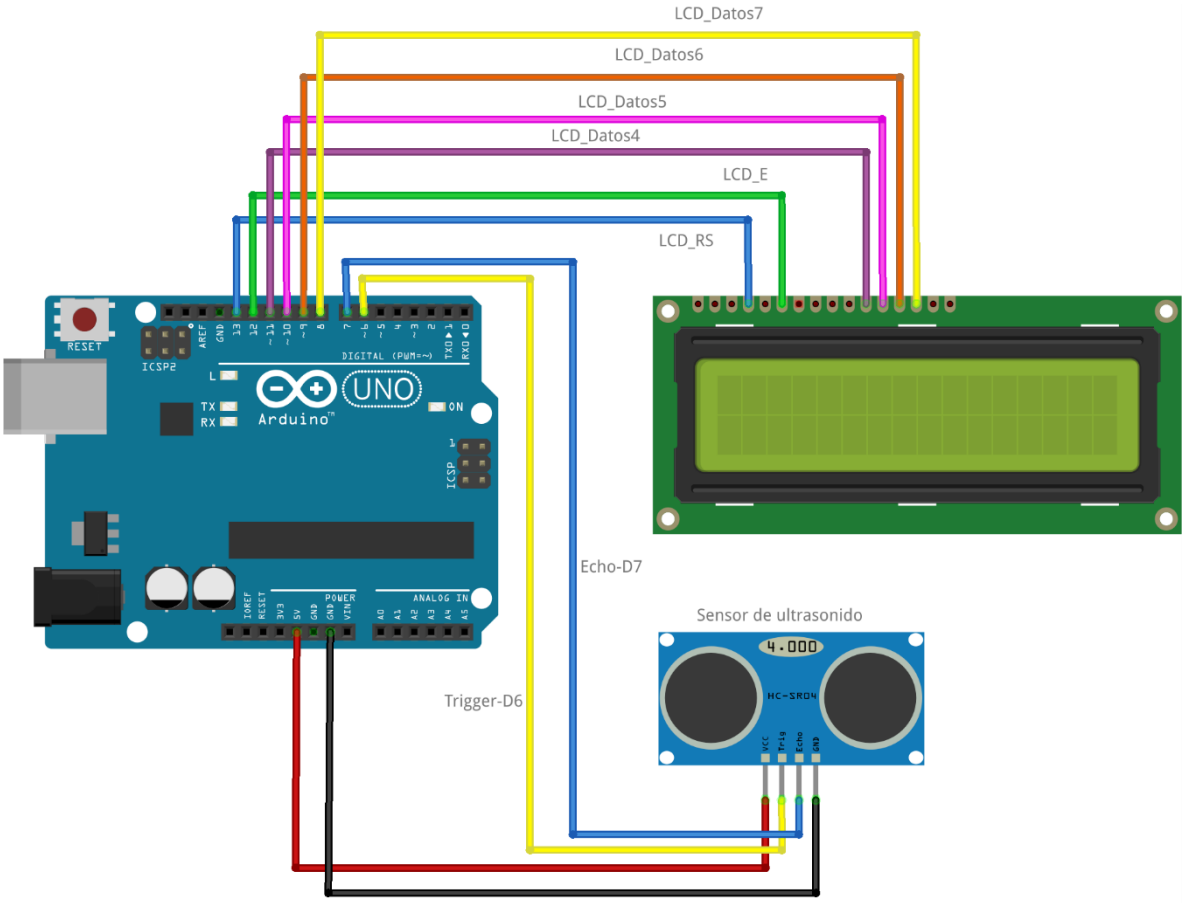


Figura 26. Diagrama de conexión del sensor ultrasónico y la pantalla LCD.

Como ejemplo de la operación del sensor ultrasónico se muestra un programa que determina la distancia a un objeto y envía el resultado por el puerto serial a la computadora. Este dato se puede visualizar utilizando la función *Serial Monitor* del ambiente de programación (IDE) del Arduino.

Tabla 15. Programa ejemplo de uso del sensor ultrasónico.

```
/*  HC-SR04
 *  Programa que lee un sensor ultrasónico y envía la distancia
 *  por el puerto serial
 *  2021
 */

#define TRIG 6   //D6
#define ECHO 7  //D7

long distancia=0;
long tiempo=0;
void setup(){
  Serial.begin(115200);
  pinMode(TRIG, OUTPUT); // Pin trigger
  pinMode(ECHO, INPUT);  // Pin echo
  digitalWrite(TRIG,LOW); //Inicializa el trigger en bajo
}

void loop(){
  digitalWrite(TRIG, HIGH); //Línea de trigger en alto
  delayMicroseconds(10);  //Espera 10 microsegundos
  digitalWrite(TRIG,LOW); //Activa el envío del pulso ultrasónico
  tiempo=pulseIn(ECHO, HIGH); //Determina el tiempo que tarda en
                              //llegar el eco
  distancia= int(0.017*tiempo); // Calcula la distancia al objeto
  Serial.print("Distancia ");
  Serial.print(distancia);
  Serial.print(" cm");
  Serial.println("");
  delay(500);
}
```

PRECAUCIÓN: No es posible utilizar al mismo tiempo el sensor ultrasónico y el servomotor, debido a que utilizan las mismas líneas de control.

2.12. Servomotor (pin D6)

Es posible conectar un servomotor de corriente directa, como el modelo SG90, directamente a un conector macho de tres pines de la tarjeta DASA. Como referencia, en la serigrafía se indica de qué lado va el cable café y el naranja (ver Figura 27). Para variar la posición del servomotor se puede utilizar un potenciómetro, tal como se muestra en el diagrama de la Figura 28. El potenciómetro deberá conectarse al borne con tornillos, tal como se menciona en el punto 2.2. El programa respectivo se incluye en la Tabla 16.

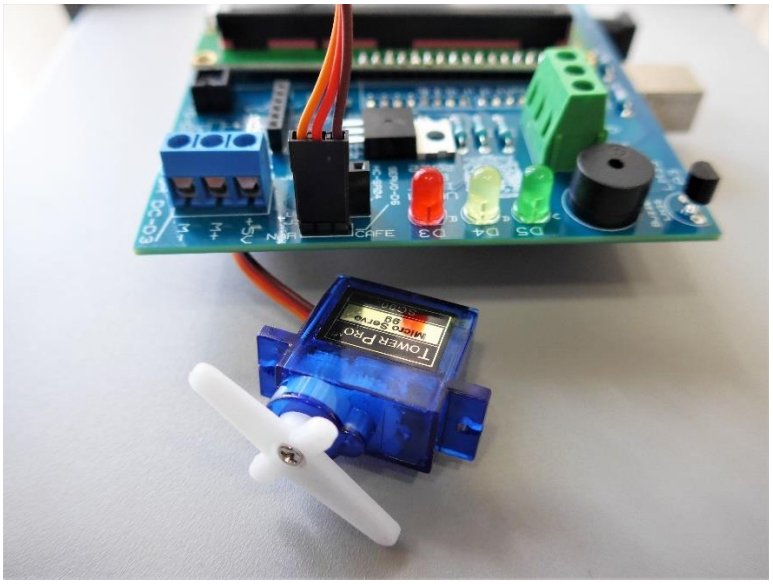


Figura 27. Conexión de un servomotor.

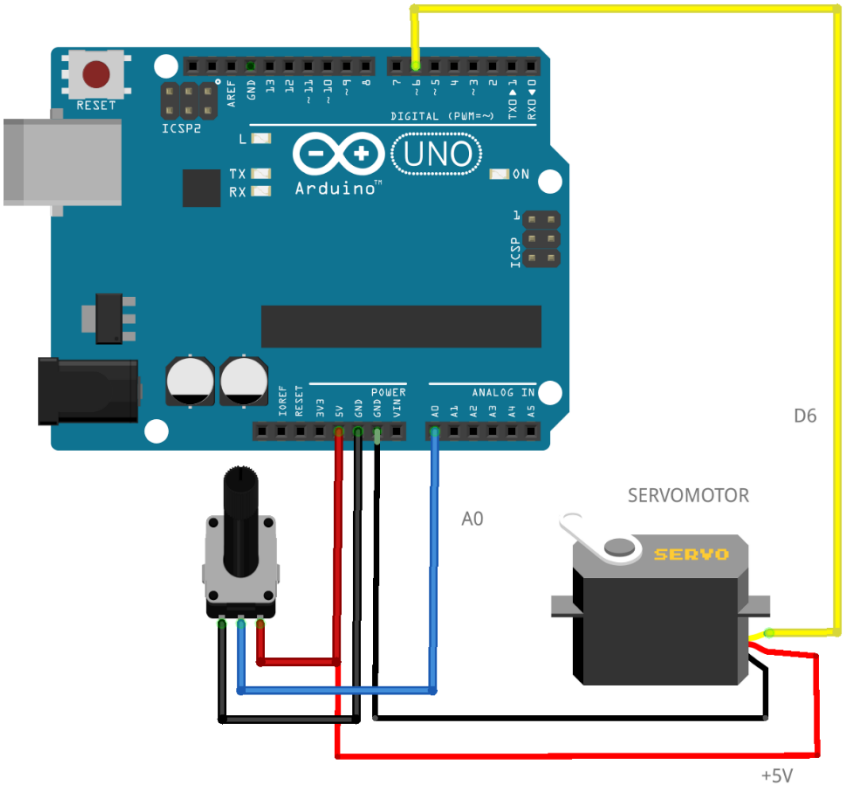


Figura 28. Conexión del servomotor y un potenciómetro.

Tabla 16. Programa para variar la posición del servomotor mediante el giro de un potenciómetro.

```
/* Servo_POT
   Programa que mueve un servo siguiendo el movimiento de un potenciómetro
   UNAM 2021
*/
#include <Servo.h>   // Biblioteca del servomotor
Servo miservo;       // Se crea un objeto "miservo"
const int pinServo = 6;    // pin del servo
const int pinPotenciometro = A0;    // pin del potenciómetro
int valPot = 0;    // Guarda el valor del potenciómetro
int angulo = 0;    // Guarda la posición angular del servo

void setup() {
  miservo.attach(pinServo);  // liga el objeto Servo al pin 6
}
void loop() {
  valPot = analogRead(pinPotenciometro);  // Lee el voltaje del potenciómetro
  angulo = map(valPot, 0, 1023, 0, 150);     // Movimiento de 0 a 150 grados
  miservo.write(angulo);     //  manda al servo la posición
  delay(15);                 //  espera a que el servo llegue a su posición
}
```

PRECAUCIÓN: No es posible utilizar al mismo tiempo el sensor ultrasónico y el servomotor, debido a que utilizan las mismas líneas de control.

2.13. Motor de corriente directa (pin D3)

El módulo DASA también cuenta con un conector para un motor de corriente directa que se pueda alimentar con los 5 V que proporciona la tarjeta Arduino UNO. El conector es un bloque de terminales con tornillo que tiene la etiqueta MOTOR DC-D3 (ver Figura 29). Mediante serigrafía se indica dónde conectar la terminal positiva del motor (M+) y la terminal negativa (M-). Si se invierte la conexión el motor girará en sentido contrario. Para controlar la velocidad del motor se utiliza una señal PWM generada mediante la línea digital D3. Esta línea está conectada a un transistor que sirve para que la corriente de operación del motor provenga de la alimentación de la tarjeta Arduino y no del pin digital D3. El diagrama de conexiones del motor y de un potenciómetro que se puede utilizar para variar la velocidad se muestra en la Figura 30. El código utilizado se muestra en la Tabla 17. La línea D5 también está conectada al LED rojo, por lo que, al variar la velocidad del motor, también cambiará la intensidad de encendido de este LED.

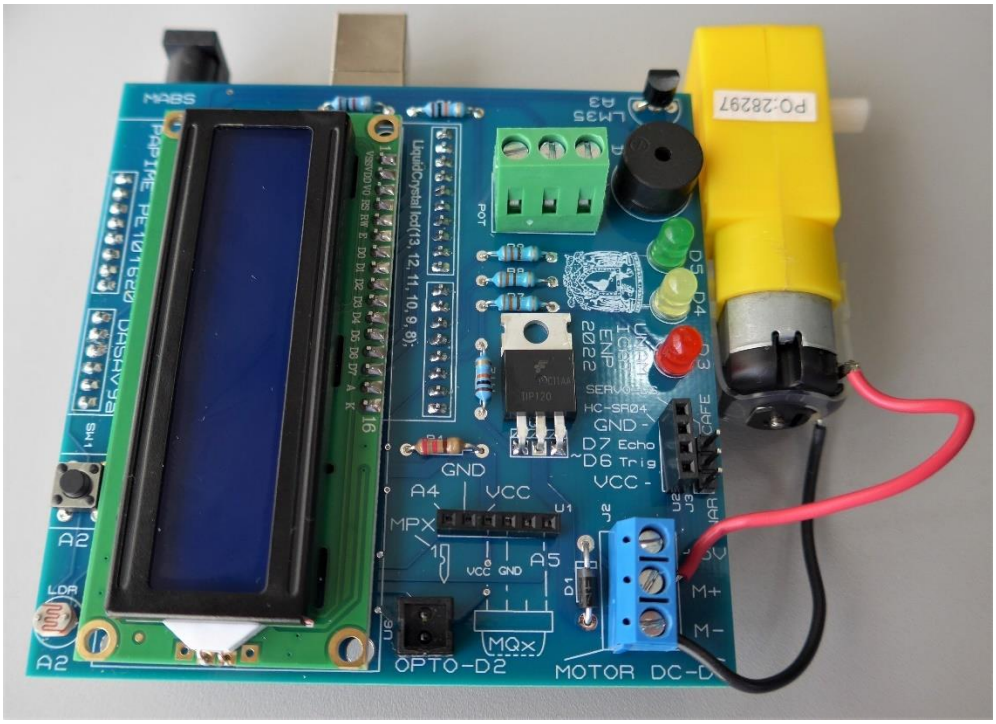


Figura 29. Conexión del motor de corriente directa.

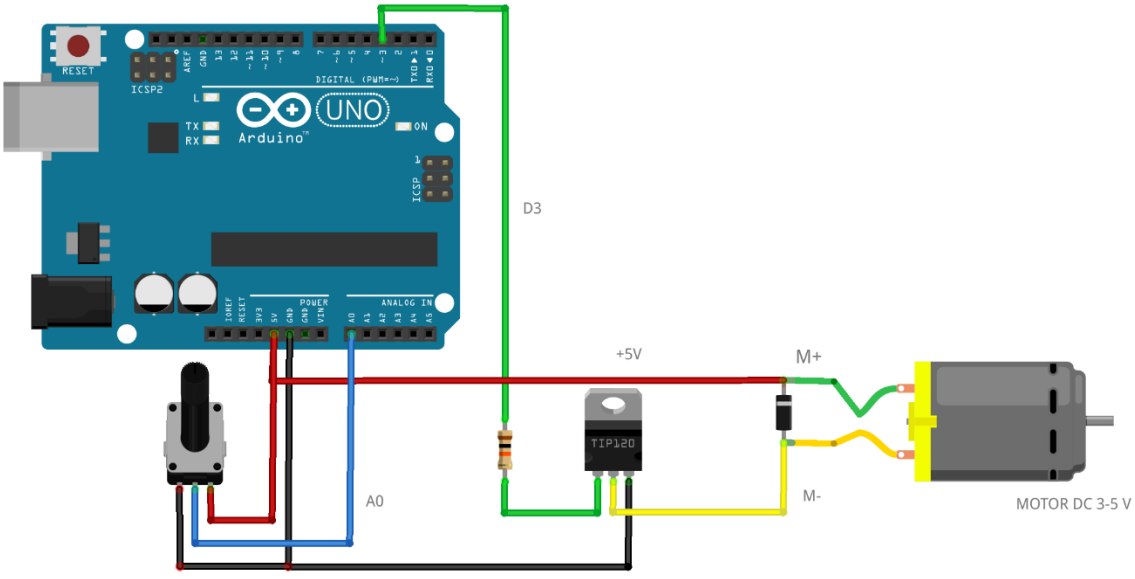


Figura 30. Diagrama de conexiones del motor de C.D. y un potenciómetro.

Tabla 17. Programa que controla la velocidad de un motor de CD con un potenciómetro.

```
/* POT_motorDC
   Programa controla la velocidad de un motor de CD
   mediante un potenciómetro
         El motor está conectado a una señal de PWM (D3)
   UNAM 2021
         */
const int pinSensor = 0;    // Sensor analógico conectado al potenciómetro
const int pinLed    = 3;    // Salida PWM conectada a D3
int brilloLed          = 0;  // Almacena el valor de brillo que se dará al LED
int valorPotenciometro = 0;  // Almacena el valor leído del potenciómetro

void setup() {
  pinMode(pinLed, OUTPUT);     // Define al pin 6 como salida PWM
}

void loop(){
  valorPotenciometro = analogRead(pinSensor);
  brilloLed = map(valorPotenciometro, 0, 1023, 0, 255);  // Reescala el valor
  analogWrite(pinLed, brilloLed);  // Se ajusta la salida del pin PWM
  delay(100);      // Retardo de 100 milisegundos
}
```

2.14. Sensor de humedad y temperatura DHT-11 (conexión externa)

Es posible conectar otros dispositivos al módulo DASA, aun cuando no se cuente con un conector específico para ello. Las terminales de los conectores proporcionan voltaje de alimentación (VCC=5 V), tierra (GND) y acceso a líneas digitales y analógicas. Todo ello está claramente indicado en la serigrafía.

Como ejemplo de utilización de un sensor que no cuenta con conexión directa al módulo DASA, utilizaremos el sensor de humedad y temperatura DHT-11. Este sensor requiere de alimentación, conexión a tierra y una línea digital de datos. Utilizando como auxiliar una pequeña protoboard, es posible conectar el sensor al conector hembra que se utiliza con el sensor de ultrasonido. El diagrama de conexiones se muestra en la Figura 31, y una fotografía del montaje físico en la Figura 32. Se puede observar que se utiliza el conector *header* hembra etiquetado como HC-SR04, mediante los pines etiquetados como VCC, GND y D6. Un programa que muestra en la pantalla LCD el dato de la temperatura en grados Celsius y el porcentaje de humedad relativa (HR), se muestra en la Tabla 18.

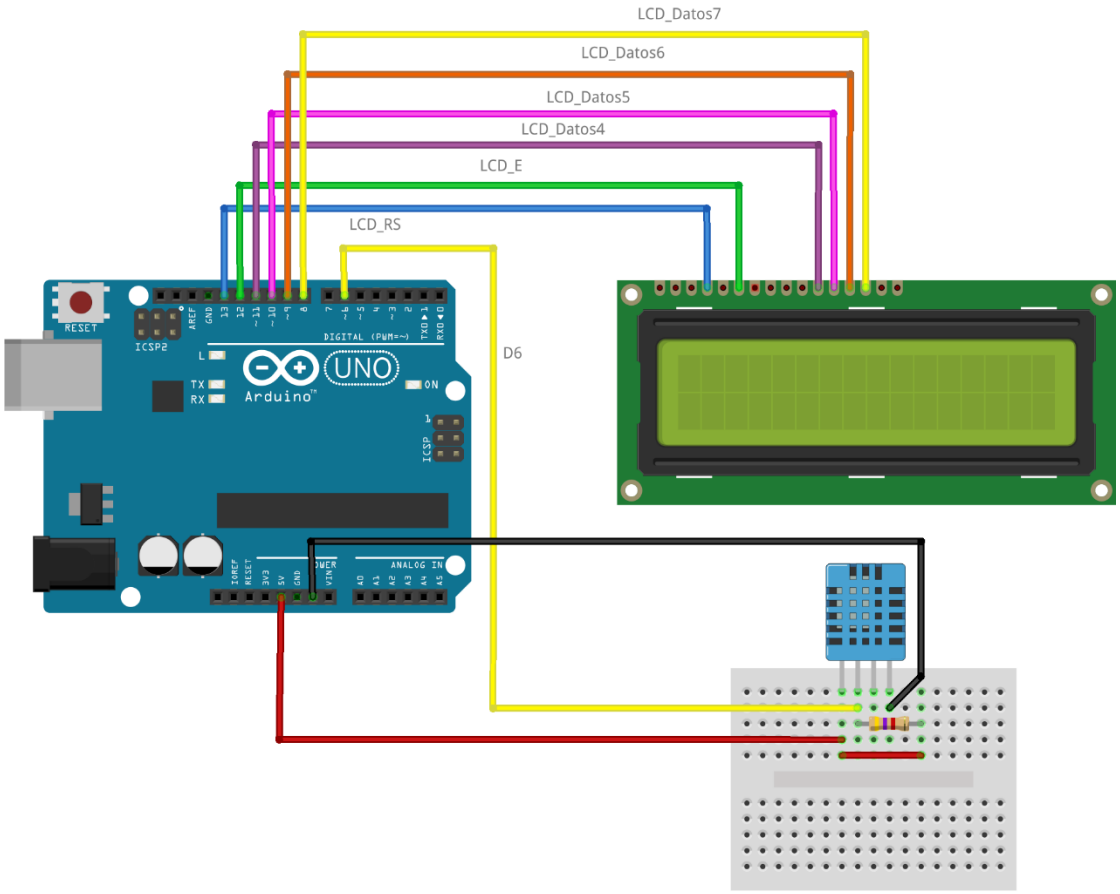


Figura 31. Conexión externa de un sensor DHT-11, utilizando una protoboard.

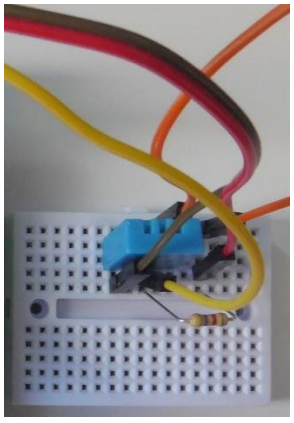


Figura 32. Montaje de un sensor DHT-11.

Tabla 18. Programa para el sensor DHT-11.

```
/* LCD_DHT11
 * Programa que lee un sensor DHT11
 * y despliega los datos en el LCD
 * Basado en el código de ejemplo
 * UNAM 2021
 * Se requiren las bibliotecas disponibles en:
 * https://github.com/adafruit/Adafruit_Sensor
 * https://github.com/adafruit/DHT-sensor-library
 *
 */

#include <LiquidCrystal.h>
#include <DHT.h>
#define DHTPIN 6     // Datos del sensor en D6
#define DHTTYPE DHT11   // DHT 11
DHT dht(DHTPIN, DHTTYPE);
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);

void setup() {
  lcd.begin(16, 2);   // LCD es de 2 renglones y 16 caracteres
  dht.begin();
  lcd.setCursor(0,0);   // Primer renglón
  lcd.print("Temp=");
  lcd.setCursor(0,1);    // Segundo renglón
  lcd.print("HR=");
}

void loop() {
  delay(2000);  //Retardo en tre mediciónes
  int h = dht.readHumidity();
  int t = dht.readTemperature();  //Lee Celsius
  lcd.setCursor(5,0);
  lcd.print(t);
  lcd.write(char(223));
  lcd.print("C");
  lcd.setCursor(3,1);
  lcd.print(h);
  lcd.print("%");
}
```

El sensor de temperatura y humedad que aparece en la Figura 32 está montado en un pequeño módulo. Es MUY IMPORTANTE notar que las conexiones son distintas a las del sensor independiente. Las conexiones del módulo se indican en la Figura 33. Se debe recordar el sensor DHT11 requiere una resistencia conectada entre la línea de datos y el voltaje de alimentación de +5 V. En el caso de utilizar el módulo, la resistencia ya está incluida. En la Figura 34, se muestra una fotografía de la conexión del módulo y su operación con la tarjeta DASA. El diagrama de conexiones para esta variante se presenta en la Figura 35.

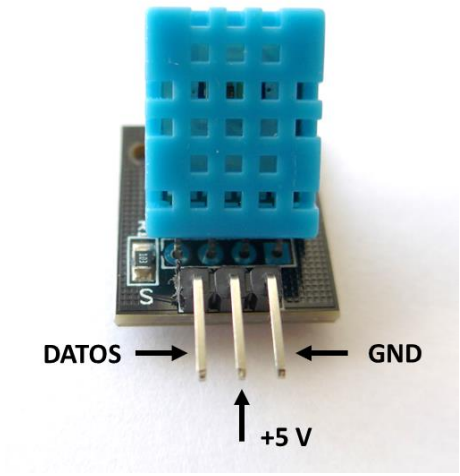


Figura 33. Conexiones del módulo sensor DHT11.

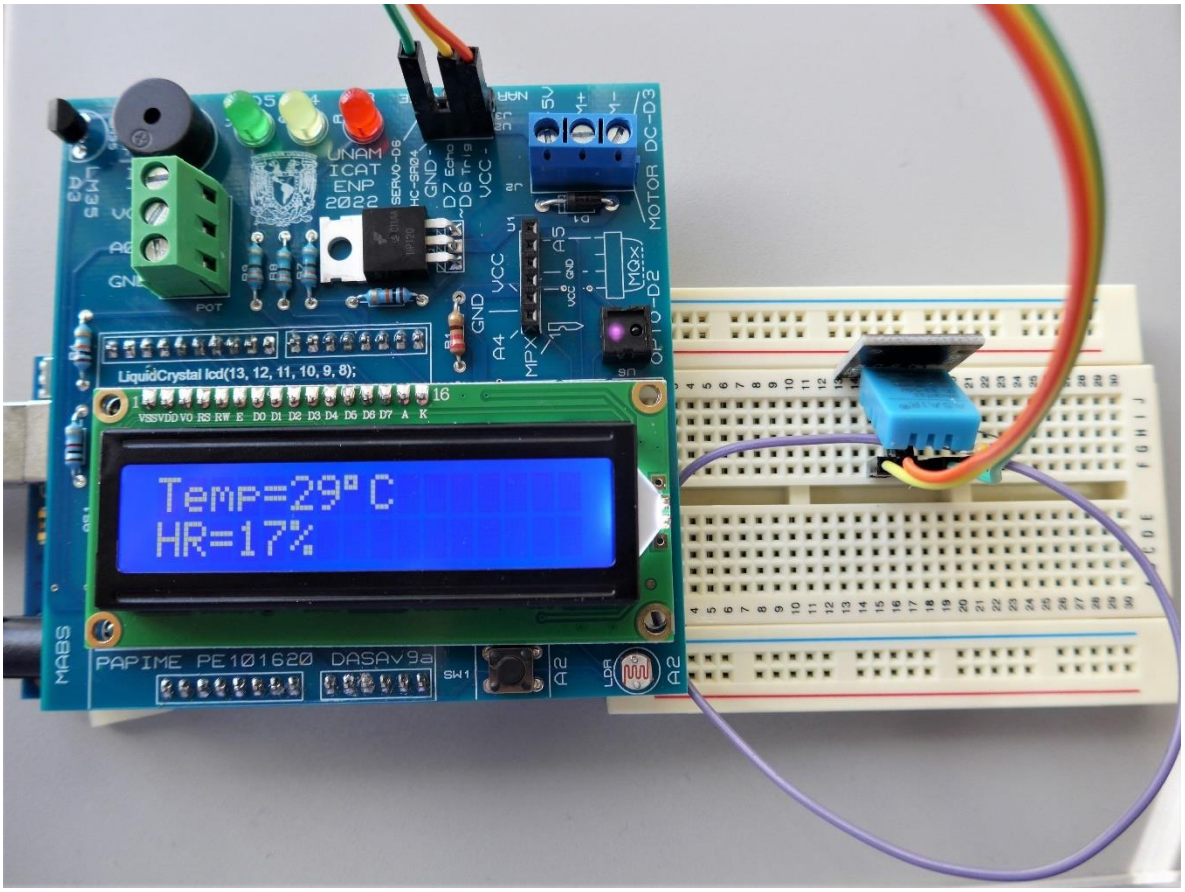


Figura 34 Conexión de un módulo DHT11 a la tarjeta DASA.

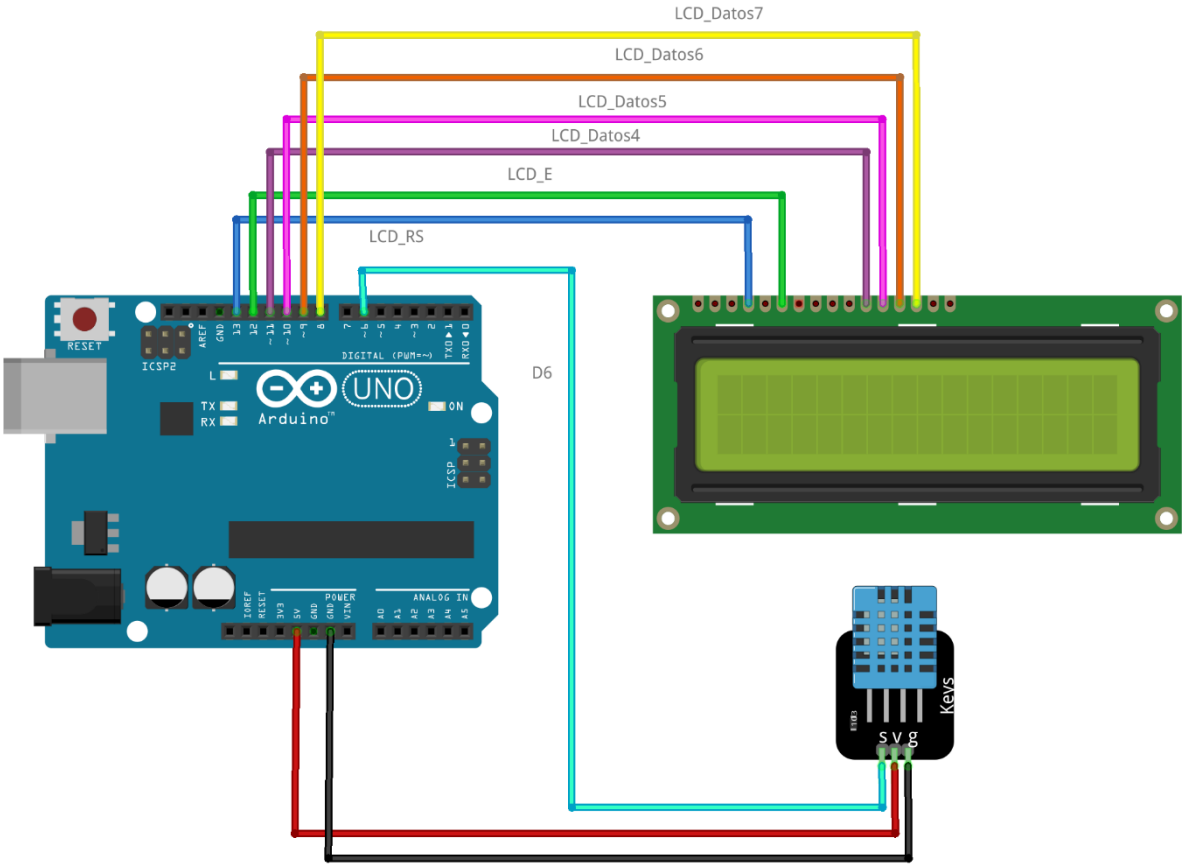


Figura 35. Diagrama de conexiones de un módulo DHT11 a la tarjeta DASA.

2.15. Sensor de presencia (PIR)

Como ejemplo de los sensores que se pueden conectar de manera externa está un sensor infrarrojo de presencia (PIR, *passive infrared*). El sensor consiste en una lente semiesférica de plástico que enfoca la radiación infrarroja en un detector. El módulo cuenta con un circuito de temporización, de manera que al detectar el movimiento de un objeto emisor de calor, se genera un pulso de duración ajustable.

En la Figura 36 se muestra el diagrama de conexión del sensor PIR. Para su conexión se utilizan las líneas de voltaje y tierra del conector header hembra que también se utiliza para conectar un sensor ultrasónico HC-SR04. En este caso se requieren tres jumpers hembra-macho. Para verificar la presencia de un objeto se utiliza el led rojo que se encuentra conectado en la línea D3 del Arduino.

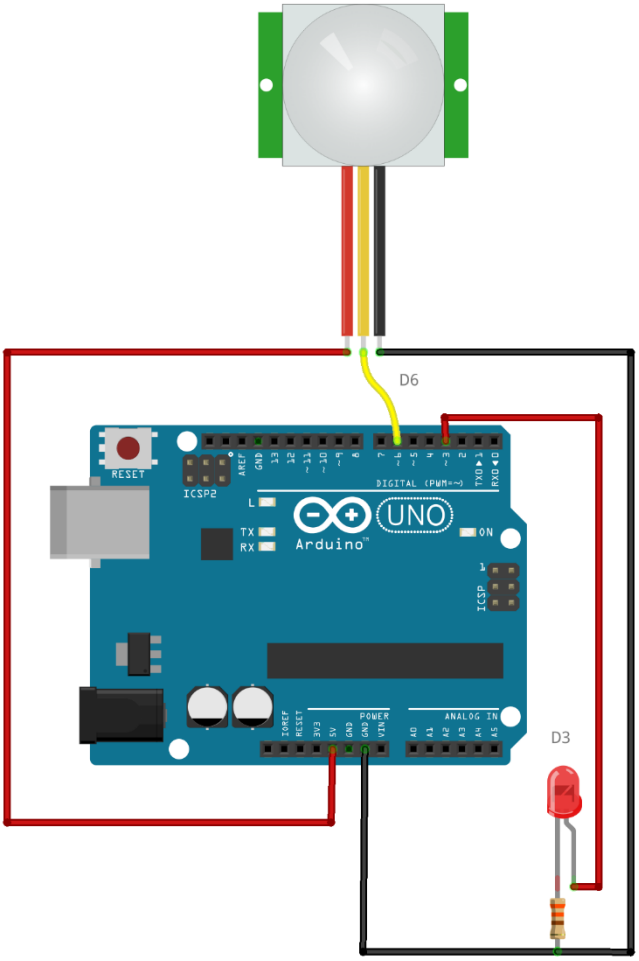


Figura 36. Diagrama de conexión de un sensor PIR y un LED.

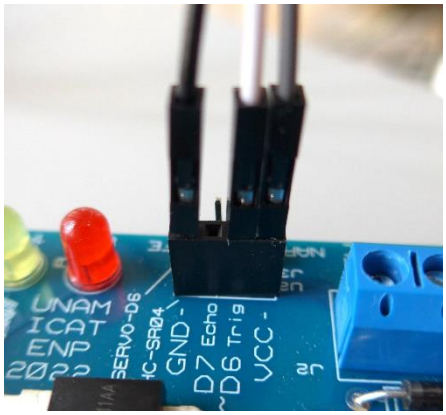


Figura 37. Detalle de la conexión externa del sensor PIR.

El modelo de sensor PIR incluido en el kit no cuenta con etiquetas en la serigrafía para identificar las conexiones, por lo que se sugiere utilizar la imagen de la Figura 38, donde además se señalan los

controles de ajuste. Se sugiere iniciar la prueba del sensor ajustando los controles al mínimo, es decir, girándolos el máximo en sentido contrario a las manecillas de un reloj. El control de sensibilidad permite determinar la distancia a la que se detectarán objetos, y el control de tiempo, la duración del pulso generado cada vez que se detecta un objeto en movimiento. Un programa de ejemplo, que enciende el led rojo de la tarjeta DASA al detectar un objeto, se muestra en la Tabla 19.

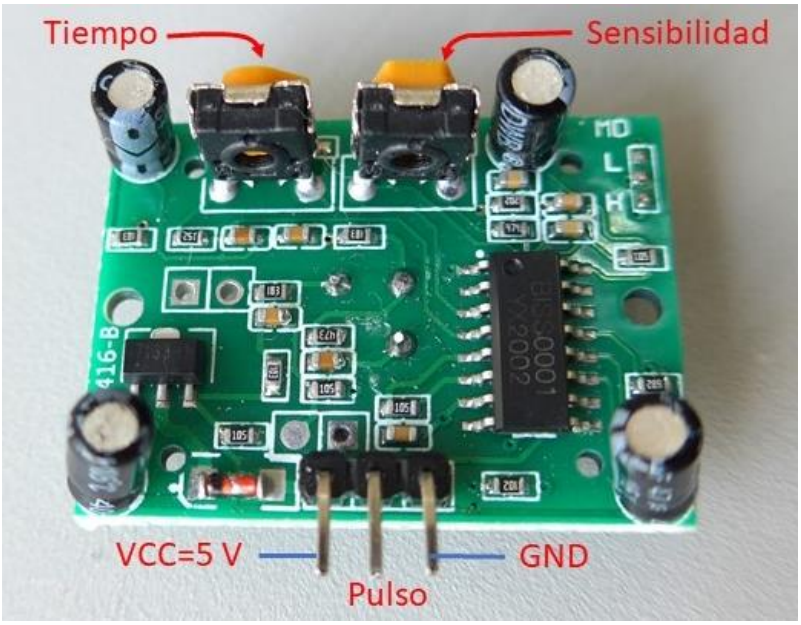


Figura 38. Pines de conexión y controles de ajuste del sensor PIR.

Tabla 19. Programa de ejemplo de uso del sensor PIR.

```
/* PIR.ino
 *  Programa que enciende un LED cuando el
 *  sensor PIR detecta presencia
 *  UNAM 2022
 */

#define LEDR 3  //led rojo en el pin 3
#define PIR 6   //Señal del PIR en el pin 6
void setup() {
  pinMode(LEDR, OUTPUT);
  pinMode(PIR, INPUT);
}

void loop() {
  if(digitalRead(PIR))
     digitalWrite(LEDR, HIGH);
  else
    digitalWrite(LEDR, LOW);
}
```

2.16. Ventilador

El kit también cuenta con un ventilador de 5 VCD como el que se muestra en la Figura 39. El ventilador es un motor de corriente directa, por lo que se puede conectar en las mismas terminales utilizadas en la sección 2.13, auxiliándose de dos jumpers hembra-hembra. La terminal roja es la terminal positiva que deberá conectarse en M+, y la terminal negra es negativa y deberá conectarse en M-. El programa de control es el mismo que se encuentra en dicha sección.

Figura 39. Imagen del ventilador de 5 VCD.

ANEXO A. Programa básico de prueba

```
/*  DASA9_test
  *  Programa que prueba el módulo v9b
  *  Enciende secuencialmente los LED
  *  Enciende el zumbador por medio segundo en dos tonos
  *  Mide y despliega la temperatura ambiente
  *  Prueba la fotoresistencia y el optointerruptor
     UNAM 2022*/
// Se requiere la biblioteca de uso del display
          #include <LiquidCrystal.h>
#define LEDR 3  //LED Rojo en el pin D3
#define LEDA 4  //LED Amarilo en el pin D4
#define LEDV 5  //LED Verde en el pin D5
#define buzzer A1  // zumbador en el pin A1
#define LM35 3  //Sensor de temperatura en pin A3
#define opto 2  //Optointerruptor en pin D2
const int fotor=A2; // pin de la fotoresistencia

// Se definen los pines que se comunican con el display
//            lcd(RS, E, D4, D5, D6, D7);
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
byte o_acento[8] = {
  0b00011,
  0b00000,
  0b01110,
  0b10001,
  0b10001,
  0b10001,
  0b01110,
  0b00000
};  //Se crea una letra o acentuada
const int pinSensor = 0;   // A0 es el pin del sensor analogico
int valorSensor = 0;
float temp = 0;
// Inicialización
void setup() {
  lcd.createChar(0, o_acento);   // Crea el caracter en la posición 0
  lcd.begin(16, 2);   // LCD es de 2 renglones y 16 caracteres

  lcd.print("M");
  lcd.write((byte)0);
  lcd.print("dulo DASA");

  pinMode(LEDR, OUTPUT);
  pinMode(LEDA, OUTPUT);
  pinMode(LEDV, OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(opto,INPUT);  //cny70
  pinMode(fotor, INPUT); //A2 como entrada digital
  tone(buzzer, 440); //440 Hz
  delay(500);
  tone(buzzer, 494);
  delay(500);
  noTone(buzzer);
  digitalWrite(LEDR, HIGH);
  delay(500);
  digitalWrite(LEDR, LOW);
  digitalWrite(LEDA, HIGH);
  delay(500);
  digitalWrite(LEDA, LOW);
  digitalWrite(LEDV, HIGH);
  delay(500);
  digitalWrite(LEDV, LOW);
}
```

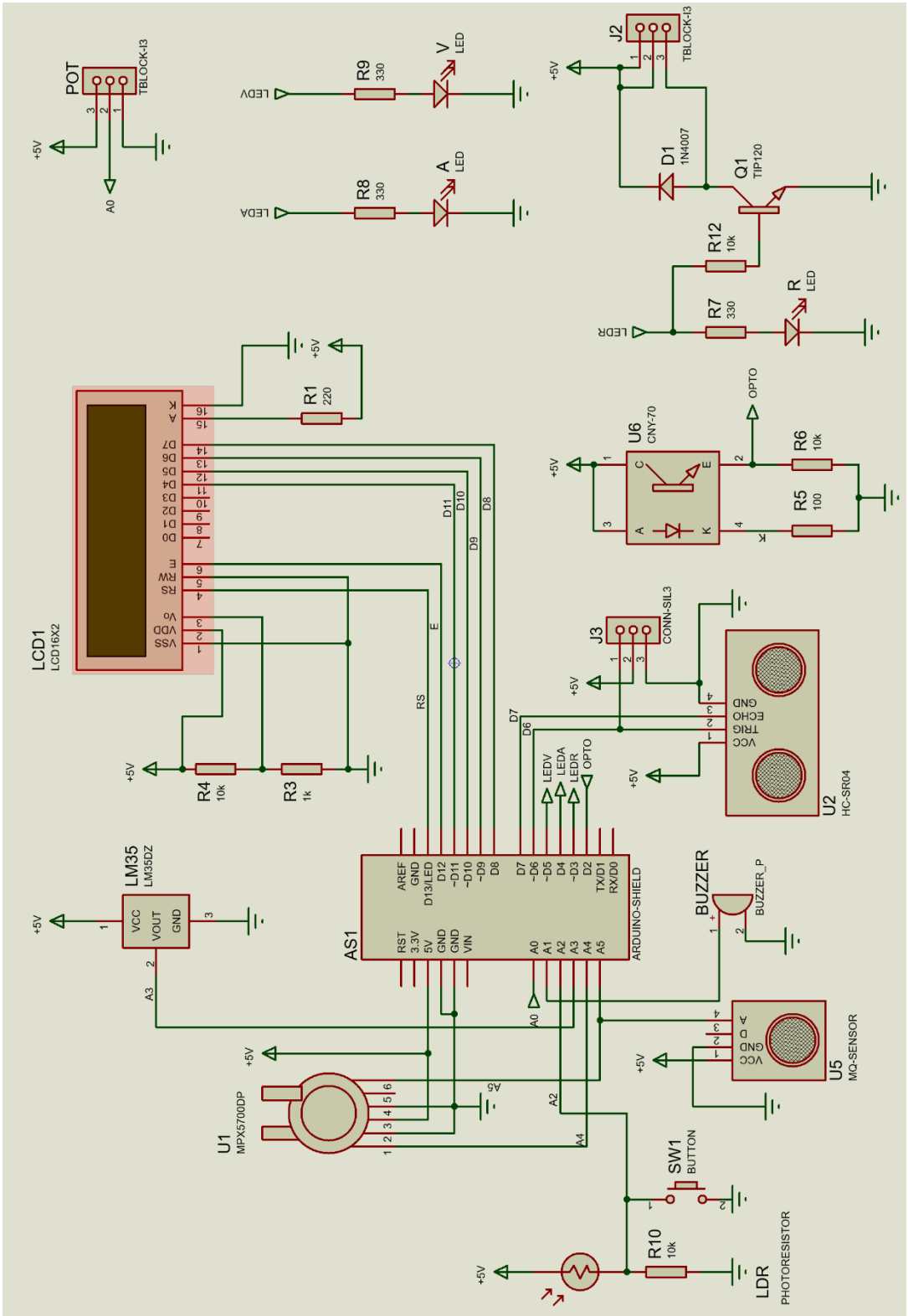
```
// Programa principal
void loop() {
  digitalWrite(LEDV, LOW);
  digitalWrite(LEDA, LOW);
  digitalWrite(LEDR, LOW);
  valorSensor = analogRead(LM35);
  temp = 500*(valorSensor/1023.0);
  lcd.setCursor(0, 1);
  lcd.print("                ");
  //lcd.setCursor(col, row)
  lcd.setCursor(0, 1);
  lcd.print("Temp= ");
  lcd.print(temp);
  lcd.print(char(223)); //Símbolo de grado
  lcd.print("C");
  delay(1000);
  lcd.setCursor(0, 1);
  lcd.print("                ");
  if(digitalRead(opto)){
    lcd.setCursor(0, 1);
    lcd.print("Opto=ALTO");
    digitalWrite(LEDR, HIGH);
  }
  else
  {
    lcd.setCursor(0, 1);
    lcd.print("Opto=BAJO");
  }
  delay(1000);
    lcd.setCursor(0, 1);
  lcd.print("                ");
  if(digitalRead(fotor)){
    lcd.setCursor(0, 1);
    lcd.print("Fotores=ALTO");

  }
  else
  {
    lcd.setCursor(0, 1);
    lcd.print("Fotores=BAJO");
    digitalWrite(LEDV, HIGH);
  }
  delay(1000);
}
```

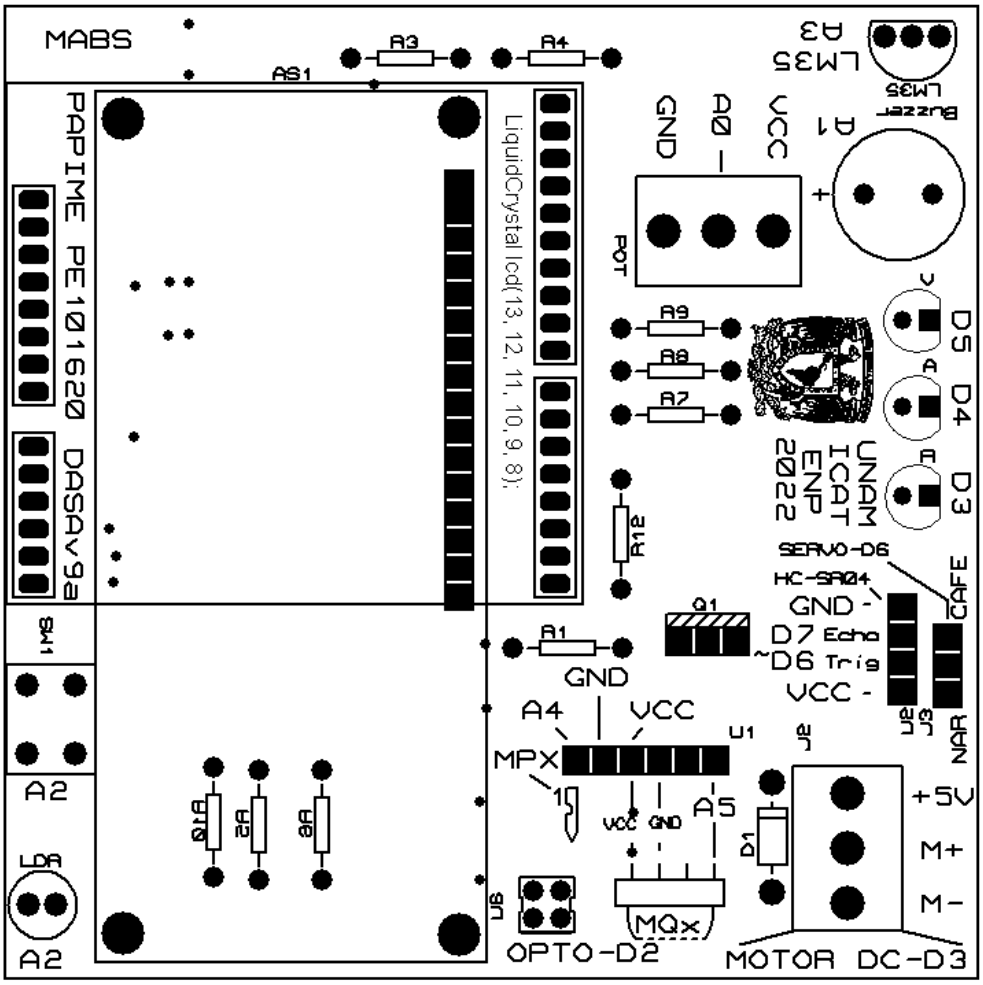
ANEXO B. Contenido del kit

Cantidad	Descripción
1	Módulo DASA
1	Tarjeta Arduino UNO
1	Cable USB
1	Protoboard 400 puntos de conexión
10	Jumpers macho-macho 20 cm
1	Potenciómetro 10 kΩ
1	Motor de CD con reductor
1	Servomotor SG90
1	Sensor ultrasónico HC-SR04
1	Sensor de temperatura y humedad HCT11
1	Sensor de presencia PIR
1	Ventilador de 5 VCD
1	Desarmador plano 1/8”x 4”
1 (opcional)	Sensor de gas MQx

ANEXO C. Diagrama esquemático



ANEXO D. SERIGRAFÍA



ANEXO E. Lista de partes del módulo DASA

Número de parte	Descripción	Cantidad
R1	Resistencia carbón 220 Ω 5% ½ W	1
R3	Resistencia carbón 1 kΩ 5% ½ W	1
R4, R6, R10, R12	Resistencia carbón 10 kΩ 5% ½ W	4
R5	Resistencia carbón 100 Ω 5% ½ W	1
R7-R9	Resistencia carbón 330 Ω 5% ½ W	3
LM35	LM35DZ Sensor de temperatura	1
U6	CNY-70 optointerruptor	1
Q1	TIP120 o TIP122. Transistor darlington NPN	1
D1	1N4007 Diodo rectificador	1
R	LED rojo 5 mm	1
A	LED amarillo 5 mm	1
V	LED verde 5 mm	1
Buzzer	Zumbador pasivo	1
LDR	Fotorresistencia 0.5 MΩ	1
J1-J2	Bloque de terminales con tornillo TRTG-03	2
LCD1	LCD 16x2	1
SW1	Push-button	1
U1	Header hembra 6 posiciones	1
U2	Header hembra 4 posiciones	1
J3	Header macho 3 posiciones	1
	Pines de conexión 16 posiciones	1
	Pines de conexión 6 posiciones	1
	Pines de conexión 8 posiciones	1
	Pines de conexión 10 posiciones	1
	Pines de conexión 10 posiciones	1
DASA v9a	Placa de circuito impreso	1