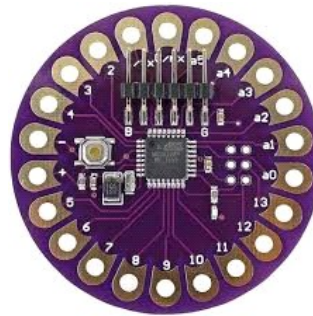
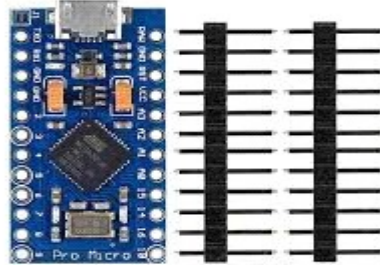
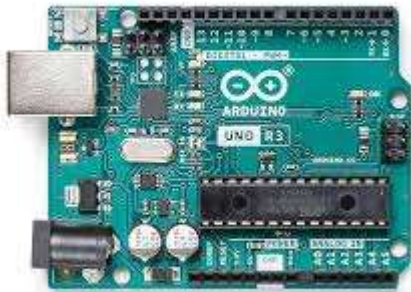


RÉVISION ARDUINO

Ardduino est une plateforme électronique open-source basée sur des microcontrôleurs. Elle permet de concevoir des projets interactifs grâce à divers composants électroniques.



Architecture d'une Carte Arduino

Chaque carte Arduino est composée de plusieurs éléments clés :

2.1. Microcontrôleur

Le cerveau de la carte, il exécute les programmes chargés dans la mémoire. Exemples :

- ATmega328P (Uno, Nano)
- ATmega2560 (Méga)
- AT91SAM3X8E (Due, basé sur ARM Cortex-M3)

2.2. Broches d'Entrée/Sortie

- **Broches numériques** : Utilisées pour les signaux HIGH ou LOW (0V ou 5V).
- **Broches analogiques** : Convertissent des signaux variables (0-1023 valeurs sur Uno).
- **PWM (Pulse Width Modulation)** : Simule une sortie analogique sur une broche numérique.
- **SPI, I2C, UART** : Protocoles de communication pour connecter des capteurs et modules.

2.3. Alimentation

- **USB (5V)** : Pour alimenter et programmer la carte via l'ordinateur.
- **Jack DC (7-12V)** : Pour une alimentation externe plus stable.
- **Broches Vin et 3.3V/5V** : Pour alimenter des composants externes.

2.4. Mémoire

- **Flash** : Stocke le programme (32 KB sur Uno).
- **SRAM** : Stocke les variables en cours d'exécution (2 KB sur Uno).
- **EEPROM** : Stocke des données non volatiles (1 KB sur Uno).

2.5. Horloge

Un oscillateur (16 MHz sur Arduino Uno) synchronise les opérations du microcontrôleur.

Composants Électroniques et Shields

Arduino peut être connecté à divers composants pour étendre ses fonctionnalités :

3.1. Capteurs et Actionneurs

Type	Exemples
Capteurs de température	DHT11, DHT22, LM35
Capteurs de distance	HC-SR04 (ultrason), IR Sharp
Capteurs de lumière	LDR (photoresistance)
Accéléromètres / Gyroscopes	MPU6050, ADXL345
Capteurs de gaz	MQ-2, MQ-135

3.2. Affichages et Communication

Type	Exemples
Afficheurs LCD	16x2, 20x4 avec I2C
OLED	SSD1306 (I2C/SPI)
Communication sans fil	Bluetooth (HC-05), Wi-Fi (ESP8266, ESP32), LoRa, RFID

3.3. Actionneurs et Moteurs

Type	Exemples
LEDs et RGB	WS2812 (NeoPixel)
Servomoteurs	SG90, MG995

Type	Exemples
Moteurs DC	L293D, TB6612FNG (pilotes moteurs)
Moteurs pas à pas	NEMA 17 avec A4988

3.4. Shields Arduino

Les shields sont des cartes d'extension qui se branchent directement sur Arduino :

- **Shield Ethernet/Wi-Fi** : Connexion à Internet.
- **Shield Motor** : Pilotage de moteurs.
- **Shield LCD Keypad** : Affichage et boutons interactifs.
- **Shield GPS** : Localisation.

Alimentation et Gestion de l'Énergie

4.1. Options d'Alimentation

- **USB (5V)** : Alimentation via ordinateur.
- **Batteries (9V, Li-Ion, Li-Po)** : Autonomie pour projets mobiles.
- **Panneaux solaires** : Pour projets autonomes.

4.2. Régulateurs de Tension

Arduino intègre des régulateurs pour éviter les surtensions :

- **Régulateur 5V** : Convertit des tensions plus élevées en 5V.
- **Régulateur 3.3V** : Alimente certains modules plus sensibles.

Conclusion

Arduino est une plateforme puissante et accessible pour l'électronique et la programmation. La maîtrise du hardware permet de créer des projets robustes et optimisés.

✦ **Prochaine étape** : Expérimenter avec des capteurs, moteurs et modules pour approfondir les connaissances.



Expérience capteur de gaz :

Dans cette expérience, nous allons utiliser un **capteur de gaz MQ-2** pour détecter la présence de gaz inflammables (butane, propane, méthane, fumée, etc.) et afficher les valeurs mesurées sur le moniteur série d'Arduino.

Matériel Nécessaire

- **Carte Arduino** (Uno, Mega, Nano...)
- **Capteur de gaz MQ-2**
- **Résistances et câbles de connexion**
- **Module buzzer** (optionnel, pour alarme sonore)
- **LED rouge** (optionnel, pour alerte visuelle)
- **Plaque d'essai** (breadboard)

Comprendre le Capteur MQ-2

Le **capteur MQ-2** détecte la présence de gaz inflammables et émet une tension proportionnelle à la concentration de gaz.

Broche	Description
VCC	Alimentation 5V
GND	Masse (0V)
A0	Sortie analogique (valeur variable selon la concentration de gaz)
D0	Sortie numérique (valeur HIGH ou LOW selon un seuil fixé)

- **Sortie analogique (A0)** : Fournit une valeur comprise entre 0 et 1023, en fonction de la concentration de gaz.
- **Sortie numérique (D0)** : Passe à HIGH (1) si le gaz détecté dépasse un seuil prédéfini, sinon reste LOW (0).



Schéma de Connexion

Branchement du capteur MQ-2 avec Arduino Uno :

- **VCC** → **5V** d'Arduino
- **GND** → **GND** d'Arduino
- **A0** → **A0** d'Arduino (pour lecture analogique)
- **Buzzer** → **D3** d'Arduino (optionnel, pour alarme sonore)
- **LED rouge** → **D4** d'Arduino (optionnel, pour alerte visuelle)

LE CODE :

```
// Définition des broches
```

```
#define MQ2_A0 A0 // Broche analogique du capteur
```

```
#define MQ2_D0 2 // Broche numérique (seuil fixé)
```

```
#define BUZZER 3 // Buzzer pour alarme
```

```
#define LED 4 // LED rouge pour alerte
```

```
void setup() {
```

```
    Serial.begin(9600); // Initialisation du moniteur série
```

```
    pinMode(MQ2_D0, INPUT); // Définition de la broche numérique en entrée
```

```
    pinMode(BUZZER, OUTPUT); // Définition de la broche du buzzer en sortie
```

```
pinMode(LED, OUTPUT); // Définition de la broche de la LED en sortie

}

void loop() {

    int valeurAnalogique = analogRead(MQ2_A0); // Lire la valeur analogique

    int etatGaz = digitalRead(MQ2_D0); // Lire la sortie numérique

    Serial.print("Valeur du capteur de gaz : ");

    Serial.println(valeurAnalogique);

    // Si la concentration de gaz est élevée, activer l'alarme

    if (etatGaz == HIGH) {

        digitalWrite(BUZZER, HIGH);

        digitalWrite(LED, HIGH);

        Serial.println("⚠ Alerte Gaz détecté ! ⚠");

    } else {

        digitalWrite(BUZZER, LOW);

        digitalWrite(LED, LOW);

    }

    delay(1000); // Attendre 1 seconde avant la prochaine mesure

}
```


*Expérience 2 : Écran LCD 16*2 + ARDUINO*

L'expérience avec un écran LCD I2C consiste à afficher du texte sur un écran LCD en utilisant le protocole de communication I2C (Inter-Integrated Circuit). L'I2C permet de réduire le nombre de fils nécessaires pour la connexion par rapport à la communication parallèle traditionnelle.

MATÉRIEL REQUIS :

- Arduino (Uno, Méga, Nano, etc.)
- Écran LCD 16x2 ou 20x4 avec module I2C intégré (PCF8574)
- Câbles de connexion (Dupont)
- Logiciel Arduino IDE



CONNEXION :

L'écran LCD avec module I2C possède généralement **4 broches** :

1. **GND** (Masse) → GND de l'Arduino
2. **VCC** (Alimentation) → 5V de l'Arduino
3. **SDA** (Données) → A4 sur Arduino Uno/Nano (ou SDA sur d'autres cartes)

4. **SCL** (Horloge) → A5 sur Arduino Uno/Nano (ou SCL sur d'autres cartes)

BIBLIOTHÈQUES NÉCESSAIRES :

Arduino ne reconnaît pas directement l'écran LCD en I2C, donc nous utilisons une bibliothèque appelée **LiquidCrystal_I2C**.

1. **Installer la bibliothèque :**

- Ouvrir Arduino IDE
- Aller dans **Croquis** → **Inclure une bibliothèque** → **Gérer les bibliothèques**
- Rechercher **LiquidCrystal_I2C** et installer la bibliothèque de **Frank de Brabander**.

Le code :

```
#include <Wire.h>
```

```
#include <LiquidCrystal_I2C.h>
```

```
// Initialisation de l'écran LCD à l'adresse 0x27 avec un écran 16x2
```

```
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

```
void setup() {
```

```
    lcd.init();           // Initialisation de l'écran LCD
```

```
    lcd.backlight();      // Active le rétroéclairage
```

```
    lcd.setCursor(0, 0);  // Positionne le curseur (colonne 0, ligne 0)
```

```
    lcd.print("Bonjour !"); // Affiche le message
```

```
lcd.setCursor(0, 1); // Seconde ligne

lcd.print("I2C fonctionne");

}

void loop() {

    // Rien à exécuter en boucle

}
```