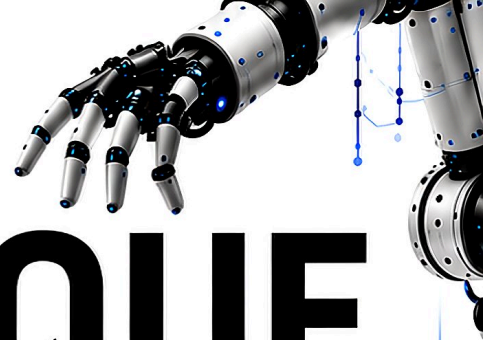




ROBOTIQUE MODERNE

GUIDE COMPLET POUR DÉBUTANTS

ARDUINO • ESP32 • IA • DRONES • CAPTEURS • PROJETS PRATIQUES



APPRENDRE

Les bases
de la robotique



COMPRENDRE

Les capteurs,
moteurs et
électronique



PROGRAMMER

Arduino & ESP32
Facile et pratique



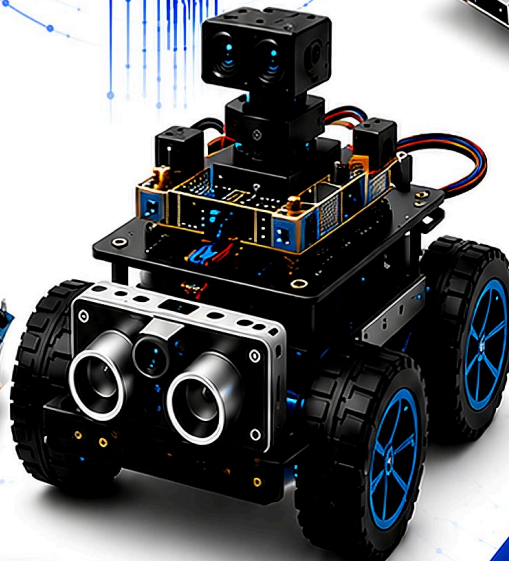
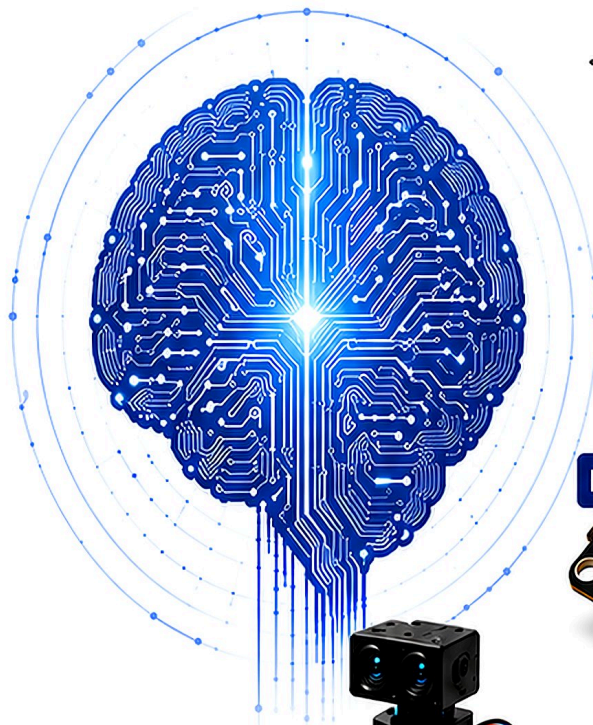
CRÉER

Des projets
réels pas à pas



INTELLIGENCE ARTIFICIELLE

L'IA au service
de la robotique



10
CHAPITRES

129 PAGES
ILLUSTRÉES



DE LA THÉORIE À LA PRATIQUE :
APPRENEZ, COMPRENEZ ET CRÉEZ VOS PROPRES ROBOTS !

COPYRIGHT

Robotique Moderne – Guide Complet pour Débutants



© 2026 Dex Atomes LLC — Tous droits réservés.

Aucune partie de cette publication ne peut être reproduite, stockée dans un système de récupération ou transmise sous quelque forme ou par quelque moyen que ce soit, électronique, mécanique, photocopie, enregistrement ou autre, sans l'autorisation écrite préalable de l'éditeur, à l'exception de courtes citations utilisées dans le cadre de critiques ou d'articles éducatifs.



Publié par : Dex Atomes LLC — Orlando, Florida, États-Unis

Dex Atomes LLC

Orlando, Florida, États-Unis



Clause de non-responsabilité

Les projets, schémas électroniques, programmes, exemples de code et instructions présentés dans ce livre sont fournis à des fins éducatives uniquement. L'éditeur n'est pas responsable de l'utilisation faite de ces projets, composants électroniques, batteries, drones, robots et équipements mentionnés. L'éditeur n'assume aucune responsabilité des dommages matériels. Respectez la législation locale sur l'utilisation des informations contenues dans cet ouvrage.



Marques commerciales

Arduino (OPEN SOURCE, Italy®), MH, sont parmi les marques citées dans ce livre et demeurent la propriété de leurs détenteurs respectifs.



Première édition : 2026



Langue : Français



Nombre de pages : 129



Site web : <https://dexatomes.com>

INTRODUCTION

Bienvenue dans le monde de la robotique !

La robotique est aujourd'hui au cœur de l'innovation technologique. Des robots industriels aux drones autonomes, en passant par les assistants intelligents et les véhicules connectés, elle transforme notre façon de vivre, de travailler et d'apprendre.

Ce livre a été conçu pour vous accompagner pas à pas, de la théorie à la pratique, afin que vous puissiez comprendre les principes fondamentaux et créer vos propres robots et systèmes intelligents, même si vous débutez.

À travers des explications claires, des schémas détaillés, des exemples de code et des projets concrets, vous découvrirez comment l'électronique, la programmation et l'intelligence artificielle se combinent pour donner vie à des machines capables d'interagir avec le monde réel.



OBJECTIF DE CE LIVRE

- ✓ Apprendre les bases de la robotique et de l'électronique.
- ✓ Maîtriser Arduino, ESP32 et les principaux composants.
- ✓ Comprendre les capteurs, les actionneurs et les systèmes embarqués.
- ✓ Explorer l'intelligence artificielle et la vision par ordinateur.
- ✓ Construire des projets pratiques et motivants.



CE QUE VOUS ALLEZ APPRENDRE



Électronique
et composants
essentiels



Programmation
avec Arduino
et ESP32



Capteurs,
actionneurs
et moteurs



Conception de
robots mobiles
et bras robotisés



Intelligence
artificielle et
vision par ordinateur

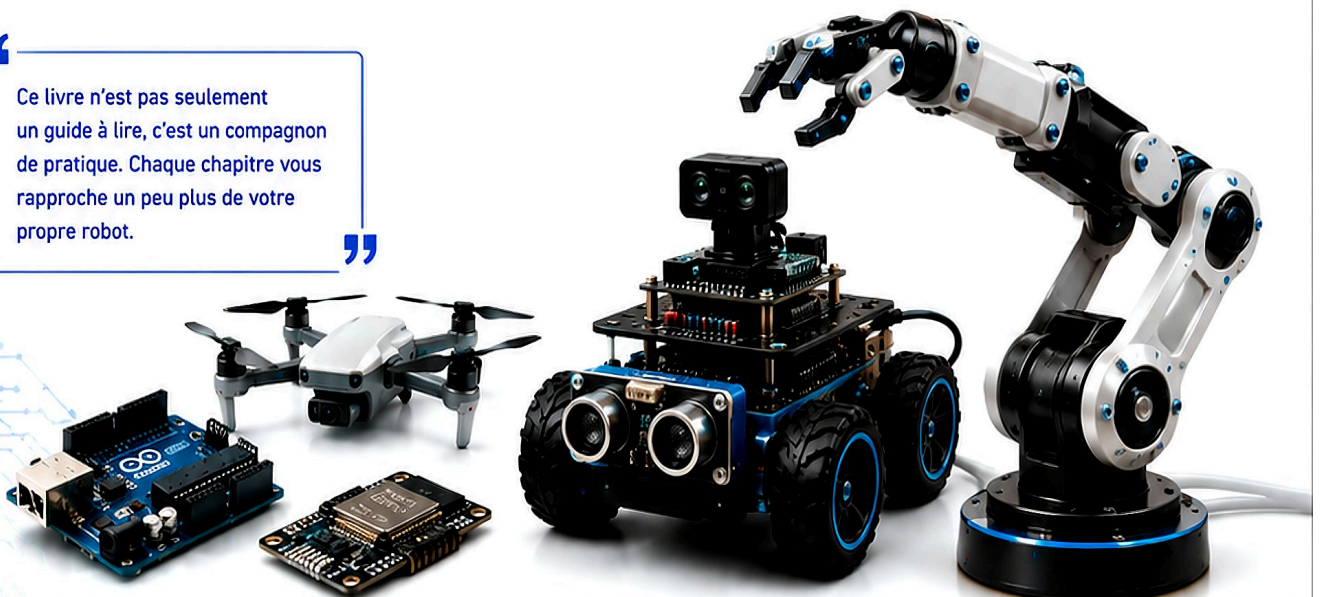


Drones et
systèmes
autonomes

“

Ce livre n'est pas seulement un guide à lire, c'est un compagnon de pratique. Chaque chapitre vous rapproche un peu plus de votre propre robot.

”



Que vous soyez étudiant, maker ou passionné de technologie, ce livre vous donnera les compétences et la confiance nécessaires pour créer, innover et bâtir le futur.

COMMENT UTILISER CE LIVRE

Votre guide pratique pour apprendre la robotique pas à pas

Ce livre a été conçu pour vous accompagner de zéro à la création de projets robotiques avancés. Pour tirer le meilleur parti de son contenu, suivez ces recommandations.



1 LISEZ DANS L'ORDRE

Les chapitres sont organisés de manière progressive. Nous vous recommandons de les suivre dans l'ordre pour bien comprendre chaque concept.



2 RÉALISEZ LES EXERCICES

Chaque chapitre contient des exercices et activités pratiques pour renforcer vos acquis. Essayez-les par vous-même avant de consulter les solutions.



3 CONSTRUISEZ LES PROJETS

La meilleure façon d'apprendre la robotique est de construire. Chaque projet vous aidera à appliquer ce que vous avez appris dans des situations concrètes.



4 EXPLOREZ LE CODE

Tous les exemples de code sont fournis et expliqués étape par étape. Modifiez-les, testez-les et comprenez-les.



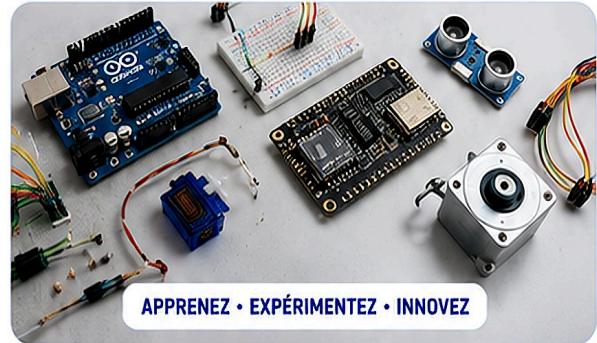
5 RÉFLÉCHISSEZ ET AMÉLIOREZ

Ne vous contentez pas de reproduire les projets. Posez des questions, cherchez des améliorations et expérimentez de nouvelles idées.



6 SOYEZ PERSÉVÉRANT

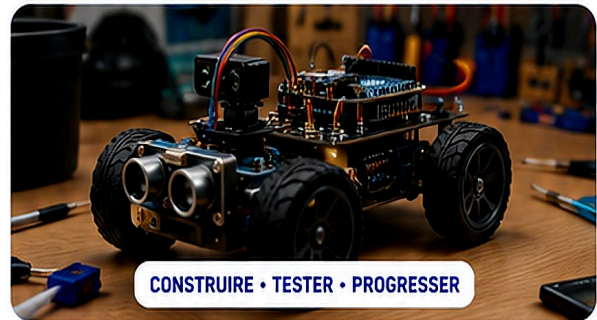
La robotique demande de la pratique et de la patience. Les erreurs font partie de l'apprentissage. Continuez, vous allez progresser !



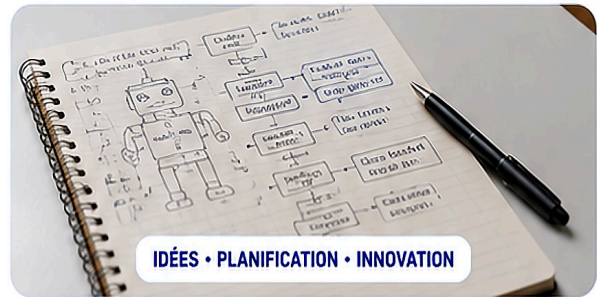
APPRENEZ • EXPÉRIMENTEZ • INNOVEZ



COMPRENDRE • CODER • TESTER



CONSTRUIRE • TESTER • PROGRESSER



IDÉES • PLANIFICATION • INNOVATION



CONSEIL IMPORTANT

Prenez des notes, documentez vos projets et conservez vos codes. Cela vous aidera à mieux apprendre et à créer vos propres robots uniques.



TABLE DES MATIÈRES

PARTIE 1 / 2

Copyright	2
Introduction	3
Comment utiliser ce livre	4
Table des matières	5–6
Ce que vous allez construire	7
Conventions et icônes utilisées	8



1

INTRODUCTION À LA ROBOTIQUE 9

Qu'est-ce que la robotique ? Histoire, domaines d'application et avenir de la robotique.



2

ÉLECTRONIQUE FONDAMENTALE 21

Les composants électroniques, lois fondamentales, circuits de base et outils de mesure.



3

ARDUINO POUR LA ROBOTIQUE 33

Découverte d'Arduino, programmation, entrées/sorties et projets pratiques.



4

ESP32 ET OBJETS CONNECTÉS 45

Programmation de l'ESP32, Wi-Fi, Bluetooth, IoT et communication en temps réel.



5

CAPTEURS ET ACQUISITION DE DONNÉES 57

Types de capteurs, acquisition de données et traitement des informations.

TABLE DES MATIÈRES

PARTIE 2 / 2

6

ACTIONNEURS ET MOTEURS 69

Moteurs DC, servomoteurs, moteurs pas-à-pas
et contrôle de mouvement.


7

CONSTRUCTION D'UN ROBOT MOBILE 81

Châssis, assemblage mécanique, câblage
et premiers tests de déplacement.


8

INTELLIGENCE ARTIFICIELLE ET VISION 93

Reconnaissance d'images, machine learning embarqué
et vision par ordinateur.


9

DRONES ET SYSTÈMES AUTONOMES 103

Architecture des drones, contrôle de vol,
stabilisation et autonomie.


10

PROJET FINAL DE ROBOTIQUE 109

Robot complet : conception, assemblage,
programmation et démonstration finale.



ANNEXES

Glossaire des termes techniques 125

Ressources recommandées 127

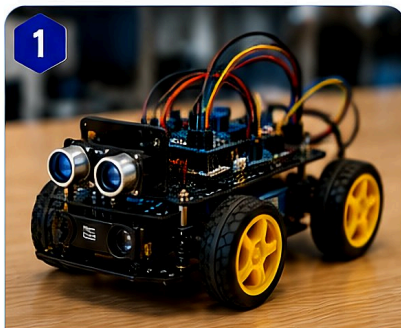
Conclusion 128



CE QUE VOUS ALLEZ CONSTRUIRE

Tout au long de ce livre, vous allez apprendre en construisant.

Chaque chapitre vous guide pas à pas dans la réalisation de projets concrets, qui vous permettront d'acquérir des compétences solides et pratiques.



ROBOT MOBILE

Construisez un robot capable de se déplacer, d'éviter les obstacles et de suivre des lignes. Vous apprendrez la programmation, le contrôle des moteurs et l'intégration des capteurs.



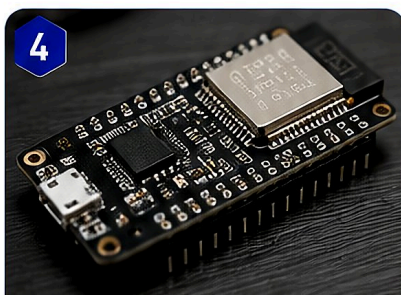
BRAS ROBOTISÉ

Créez un bras robotisé à plusieurs degrés de liberté. Apprenez à contrôler des servomoteurs et à exécuter des mouvements précis pour saisir et déplacer des objets.



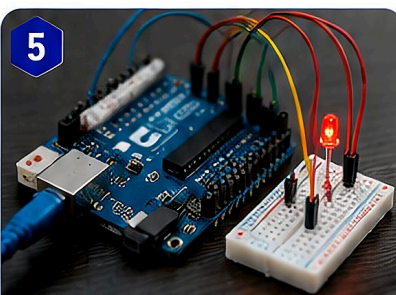
DRONE AUTONOME

Découvrez les principes du vol et construisez un drone capable de décoller, se stabiliser et effectuer des missions de base de manière autonome.



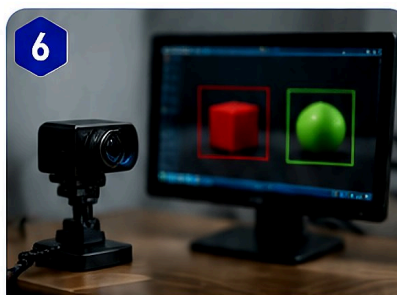
PROJETS AVEC ESP32

Utilisez la puissance de l'ESP32 pour créer des projets connectés : Wi-Fi, Bluetooth, Internet des objets (IoT) et communication en temps réel. Envoyez et recevez des données depuis le monde entier.



PROJETS AVEC ARDUINO

Maîtrisez les bases d'Arduino en réalisant des circuits interactifs : commandes de LED, boutons, capteurs, affichages et plus encore. Idéal pour débiter en électronique et en programmation.



VISION PAR ORDINATEUR

Initiez-vous à l'intelligence artificielle et à la vision par ordinateur. Apprenez à détecter, reconnaître et suivre des objets à l'aide de votre caméra et de Python.



OBJECTIF FINAL

À la fin de ce livre, vous aurez les compétences et la confiance nécessaires pour concevoir et construire vos propres systèmes robotiques innovants.



CONVENTIONS ET ICÔNES UTILISÉES

Pour rendre votre apprentissage plus clair et interactif, ce livre utilise des icônes et des encadrés spéciaux tout au long du contenu.
Voici la signification de chaque icône et comment les utiliser efficacement.



💡 ASTUCE

Cette icône vous indique des conseils pratiques, des raccourcis, des bonnes pratiques ou des recommandations tirées de l'expérience. Lisez-les attentivement, ils peuvent vous faire gagner du temps !



⚠️ ATTENTION

Indique un point important à ne pas négliger. Cela peut être une erreur fréquente, un risque matériel, ou une règle de sécurité à respecter.



🔧 PROJET PRATIQUE

Signale un projet ou une activité pratique à réaliser. Chaque projet vous permet d'appliquer les connaissances acquises et de construire quelque chose de concret.



💻 CODE

Indique un exemple de code ou un extrait de programme. Ces sections incluent des explications détaillées pour comprendre chaque ligne et savoir comment l'utiliser.



🧠 CONCEPT IMPORTANT

Met en évidence une notion clé ou fondamentale à bien comprendre. Prenez le temps de bien assimiler ces concepts, car ils sont essentiels pour progresser.



📌 À RETENIR

Résume l'essentiel d'un chapitre ou d'une section. Relisez ces encadrés pour mémoriser les points importants avant de passer au chapitre suivant.



COMMENT UTILISER CES ICÔNES ?

Lorsque vous voyez l'une de ces icônes, prenez le temps de lire attentivement le contenu associé. Elles sont là pour vous guider, vous alerter ou vous aider à mieux comprendre et appliquer ce que vous apprenez. Elles rendent votre apprentissage plus efficace et agréable.



CHAPITRE 1

INTRODUCTION À LA ROBOTIQUE

La robotique est la combinaison de plusieurs disciplines telles que la mécanique, l'électronique, l'informatique et l'intelligence artificielle pour concevoir et construire des machines capables d'effectuer des tâches de manière autonome ou semi-autonome.

Dans ce chapitre, vous découvrirez ce qu'est la robotique, son histoire, ses domaines d'application et les compétences de base nécessaires pour commencer votre apprentissage.

DANS CE CHAPITRE, VOUS ALLEZ DÉCOUVRIR :



1. **Qu'est-ce que la robotique ?**
Définition et concepts clés.



2. **Histoire de la robotique**
De l'Antiquité à la robotique moderne.



3. **Domaines d'application**
Où trouve-t-on des robots dans le monde réel ?



4. **Types de robots**
Robots industriels, mobiles, humanoïdes et plus.



5. **Compétences de base**
Les connaissances essentielles pour débiter.



6. **Mini-projet d'exploration**
Votre premier pas pratique en robotique.

“

La robotique ne remplace pas l'humain, elle augmente ses capacités.

— L'avenir appartient à ceux qui conçoivent l'avenir.

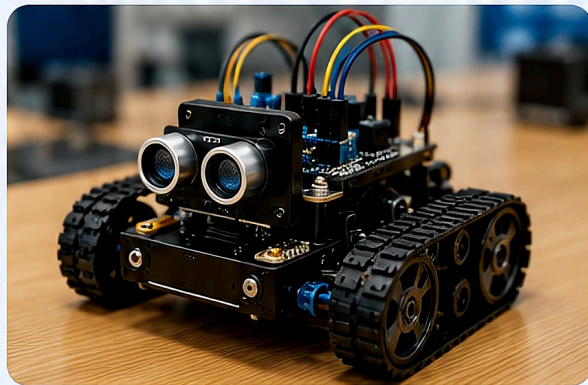
”



OBJECTIFS DU CHAPITRE

À la fin de ce chapitre, vous serez capable de :

- ✓ Expliquer ce qu'est la robotique et pourquoi elle est importante.
- ✓ Connaître les principales étapes de son évolution.
- ✓ Identifier les différents domaines où les robots sont utilisés.
- ✓ Reconnaître les principaux types de robots.
- ✓ Comprendre les compétences nécessaires pour devenir un roboticien.
- ✓ Réaliser un mini-projet simple pour appliquer vos premières connaissances.



ASTUCE :

Lisez attentivement chaque section et essayez les activités pratiques.
La pratique régulière est la clé pour maîtriser la robotique !



1.1 QU'EST-CE QUE LA ROBOTIQUE ?



DÉFINITION

La robotique est une discipline scientifique et technologique qui combine la **mécanique**, l'**électronique**, l'**informatique** et le **contrôle** pour concevoir, construire et utiliser des **robots**.

Elle permet de créer des machines capables d'effectuer des tâches de manière autonome ou semi-autonome dans le monde réel.



QU'EST-CE QU'UN ROBOT ?

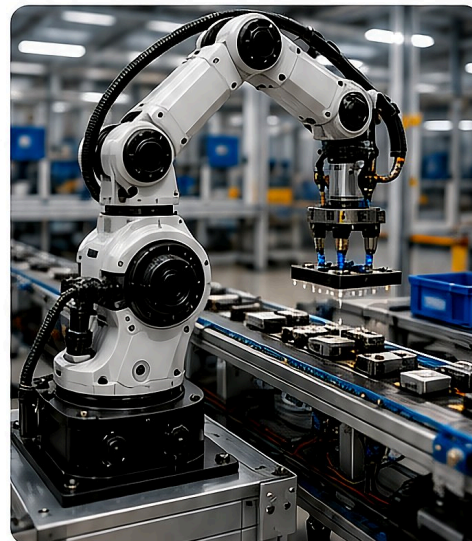
Un **robot** est une machine programmable capable de percevoir son environnement, de traiter les informations, de prendre des décisions et d'agir sur le monde physique afin d'accomplir une ou plusieurs tâches spécifiques.

Il peut fonctionner de façon autonome ou être commandé par un être humain à distance.

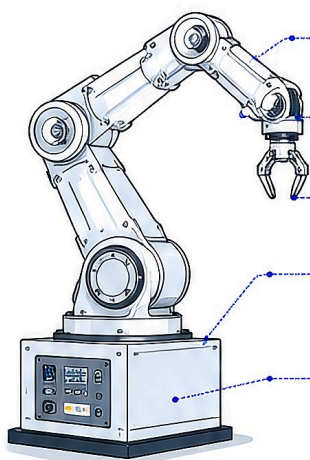


ASTUCE

Pensez à un robot comme à un être vivant artificiel : il possède des sens (capteurs), un cerveau (contrôleur) et des muscles (actionneurs).



LES COMPOSANTS PRINCIPAUX D'UN ROBOT



CAPTEURS

Collectent des informations sur l'environnement (position, distance, température, lumière, etc.).

CONTRÔLEUR

Traite les informations reçues, prend des décisions et envoie des commandes.

ACTIONNEURS

Convertissent l'énergie en mouvement (moteurs électriques, vérins, etc.).

STRUCTURE MÉCANIQUE

Supporte et relie tous les composants. Détermine la forme et la mobilité du robot.

SOURCE D'ÉNERGIE

Fournit l'énergie nécessaire au fonctionnement du robot (batterie, alimentation, etc.).



CARACTÉRISTIQUES D'UN ROBOT

✓ PROGRAMMABLE

Un robot suit un programme pour accomplir des tâches.

✓ POLYVALENT

Il peut être conçu pour effectuer différentes tâches.

✓ PRÉCIS

Il réalise des mouvements avec une grande précision.

✓ FIABLE

Il peut travailler en continu dans des conditions difficiles.

✓ ADAPTABLE

Il peut apprendre et s'adapter à son environnement.

EXEMPLES DE TÂCHES RÉALISÉES PAR DES ROBOTS



Assemblage de pièces dans les usines



Soudure et découpe de matériaux



Tri et emballage de produits



Assistance chirurgicale



Surveillance et récolte en agriculture



Livraison et transport de marchandises



Exploration spatiale et sous-marine



Interventions en zones dangereuses



Aide dans les tâches domestiques



À RETENIR

La robotique n'est pas seulement l'étude des robots, c'est aussi l'art de concevoir des systèmes intelligents qui perçoivent, pensent et agissent pour améliorer la vie humaine. Elle est au cœur de nombreuses innovations technologiques de notre époque.



1.2

HISTOIRE DE LA ROBOTIQUE

La robotique a parcouru un long chemin depuis les premières idées de machines artificielles jusqu'aux robots intelligents d'aujourd'hui. Cette évolution résulte de découvertes scientifiques, d'innovations technologiques et de la vision de nombreux inventeurs et chercheurs.



ASTUCE

Comprendre l'histoire de la robotique nous aide à apprécier les avancées actuelles et à imaginer l'avenir.

CHRONOLOGIE DES GRANDES ÉTAPES

PÉRIODE	ÉVÉNEMENT	DESCRIPTION	
 Antiquité jusqu'au 18 ^e siècle	Automates et mécanismes	Les premières machines automatiques apparaissent dans les mythes et les légendes. Des inventeurs comme Héron d'Alexandrie conçoivent des dispositifs mécaniques étonnants.	
 1800 – 1920	Premiers dispositifs automatiques	Développement de machines à cames, d'horloges programmables et d'appareils capables d'effectuer des tâches répétitives, ouvrant la voie à l'automatisation.	
 1920 – 1940	Naissance du terme "robot"	Le mot « robot » est popularisé en 1921 par Karel Čapek dans sa pièce <i>R.U.R. (Rossum's Universal Robots)</i> . L'idée de machines travaillant à la place de l'humain prend forme dans la culture.	
 1940 – 1960	Premiers robots industriels	En 1948, William Grey Walter crée des robots mobiles expérimentaux. En 1954, George Devol invente le premier bras robotisé et fonde Unimation, ouvrant l'ère industrielle.	
 1960 – 1990	Essor de la robotique moderne	Les robots deviennent plus précis, plus rapides et plus polyvalents. Ils sont utilisés dans l'industrie, la médecine, l'exploration spatiale et la recherche scientifique.	
 1990 – Aujourd'hui	Robots intelligents et autonomes	Intelligence artificielle, apprentissage automatique et capteurs avancés permettent aux robots de percevoir, comprendre et interagir avec leur environnement de façon autonome.	

PERSONNAGES ET INVENTEURS MARQUANTS

- **Léonard de Vinci (1452–1519)** : conçoit des plans de machines mécaniques et d'automates.
- **Karel Čapek (1890–1938)** : écrivain qui introduit le terme « robot ».
- **George Devol (1912–2011)** : inventeur du premier robot industriel.
- **Joseph Engelberger (1925–2015)** : pionnier de la robotique industrielle et fondateur d'Unimation.
- **Isaac Asimov (1920–1992)** : auteur des *Trois Lois de la robotique*, fondement éthique essentiel.

LES TROIS LOIS DE LA ROBOTIQUE D'ISAAC ASIMOV



- 1 Un robot ne peut porter atteinte à un être humain ni, par inaction, permettre qu'un être humain soit exposé au danger.
- 2 Un robot doit obéir aux ordres donnés par les êtres humains, sauf si de tels ordres entrent en conflit avec la Première Loi.
- 3 Un robot doit protéger son existence dans la mesure où cette protection n'entre pas en conflit avec la Première ou la Deuxième Loi.



CONCEPT IMPORTANT

L'histoire de la robotique montre la créativité et la persévérance humaine. Chaque génération a repoussé les limites de la technologie pour créer des machines toujours plus intelligentes et utiles.



L'avenir appartient à ceux qui conçoivent les machines capables d'améliorer le monde.

— Isaac Asimov

1

CHAPITRE 1

1.3

DOMAINES D'APPLICATION DE LA ROBOTIQUE

La robotique est présente dans presque tous les aspects de notre vie. Des usines aux hôpitaux, des champs agricoles à l'espace, les robots améliorent la productivité, la précision, la sécurité et la qualité des services.

Chaque domaine d'application a ses propres besoins et contraintes, ce qui a conduit au développement de robots spécialisés capables d'accomplir des tâches spécifiques de manière efficace.



ASTUCE

La diversité des applications montre que la robotique est une technologie transversale qui apporte des solutions innovantes à des problèmes complexes dans de nombreux secteurs.

PRINCIPAUX DOMAINES D'APPLICATION

Domaine	Objectif principal	Exemples d'applications	Impact / Bénéfices
 Industrie (Manufacturier)	Automatiser les processus de production et améliorer la qualité.	- Robots de soudure et d'assemblage - Manipulation de pièces lourdes - Inspection et contrôle qualité	- Productivité accrue - Précision et répétabilité - Réduction des coûts et des délais
 Médecine et santé	Assister les professionnels de santé et améliorer les soins aux patients.	- Chirurgie assistée par robot - Prothèses et exosquelettes - Robots de livraison de médicaments	- Précision chirurgicale améliorée - Réduction des risques pour le personnel - Meilleur suivi et récupération
 Agriculture	Augmenter la productivité agricole et gérer les ressources naturelles.	- Drones pour la surveillance des cultures - Robots de récolte - Désherbage et semis automatisés	- Rendements optimisés - Réduction de l'utilisation de produits chimiques - Agriculture plus durable
 Aérospatiale	Explorer, construire et maintenir des systèmes dans des environnements extrêmes.	- Robots d'assemblage d'avions - Rovers d'exploration planétaire - Robots d'inspection de fusées et satellites	- Accès à des environnements difficiles - Sécurité des missions - Précision et fiabilité élevées
 Logistique et entrepôts	Optimiser le stockage, le tri et la livraison des marchandises.	- Robots de tri et de préparation de commandes - Véhicules autonomes (AGV/AMR) - Drones de livraison	- Rapidité des opérations - Réduction des erreurs - Chaîne d'approvisionnement efficace
 Défense et sécurité	Protéger les personnes et intervenir dans des situations à haut risque.	- Robots de déminage - Drones de reconnaissance - Robots de surveillance et d'intervention	- Protection des humains - Accès à des zones dangereuses - Intervention rapide et précise
 Domestique et services	Aider dans les tâches quotidiennes et améliorer le confort de vie.	- Aspirateurs robots - Robots tondeuses - Assistants personnels et robots éducatifs	- Gain de temps - Confort et praticité - Assistance aux personnes âgées ou à mobilité réduite

REMARQUES IMPORTANTES

Bien que les robots offrent de nombreux avantages, leur intégration doit être planifiée avec soin.

- Considérer l'impact social et l'emploi.
- Respecter les normes éthiques et la confidentialité.
- Assurer la sécurité et la maintenance régulière.
- Former les utilisateurs pour une collaboration efficace.



À RETENIR

La robotique n'est pas limitée à un seul domaine. Son potentiel réside dans sa capacité à s'adapter à des besoins variés et à travailler aux côtés des humains pour construire un avenir meilleur.

1

CHAPITRE 1

Introduction à la robotique

1.4

TYPES DE ROBOTS

CLASSIFICATION ET CARACTÉRISTIQUES

Les robots ne sont pas tous construits de la même manière. Selon la façon dont ils se déplacent, l'environnement dans lequel ils évoluent et les tâches qu'ils accomplissent, on peut les classer en plusieurs catégories.

Comprendre ces types de robots permet de mieux choisir la solution appropriée à un problème donné et d'anticiper les défis liés à leur conception et à leur utilisation.



CLASSIFICATION DES ROBOTS

Type de robot	Mobilité	Description	Exemples d'applications
 1. Robots industriels	Fixes ou semi-fixes	Conçus pour effectuer des tâches répétitives ou complexes avec une grande précision dans des environnements industriels contrôlés.	- Soudure, assemblage, peinture - Manipulation de pièces lourdes - Usinage, découpe, palettisation
 2. Robots mobiles	Mobiles	Capables de se déplacer dans leur environnement pour accomplir des tâches sans être fixés à un endroit.	- Robots de livraison (AGV/AMR) - Robots d'inspection - Robots de nettoyage
 3. Robots humanoïdes	Mobiles	Conçus pour imiter l'apparence et les mouvements des êtres humains afin d'interagir dans des environnements pensés pour l'homme.	- Recherche scientifique - Assistance, accueil, éducation - Divertissement
 4. Robots aériens (drones)	Aériens	Conçus pour voler ou évoluer dans les airs à l'aide de rotors ou d'ailes fixes.	- Prises de vues aériennes - Surveillance et cartographie - Livraison, agriculture
 5. Robots sous-marins (ROV/AUV)	Aquatiques	Conçus pour explorer ou travailler sous l'eau, câblés (ROV) ou autonomes (AUV).	- Exploration océanique - Inspection de structures sous-marines - Recherche scientifique
 6. Robots collaboratifs (cobots)	Fixes ou mobiles	Conçus pour travailler directement avec des humains dans un espace partagé en toute sécurité.	- Assemblage léger - Assistance sur postes de travail - Petites et moyennes entreprises

CARACTÉRISTIQUES PRINCIPALES

Malgré leurs différences, la plupart des robots partagent des caractéristiques fondamentales :

- **Capteurs** : ils perçoivent leur environnement (vision, son, température, force, etc.).
- **Actionneurs** : ils produisent un mouvement ou une force (moteurs électriques, vérins, etc.).
- **Contrôleur** : il traite les informations et prend des décisions selon un programme ou grâce à l'intelligence artificielle.
- **Source d'énergie** : elle fournit l'électricité ou la puissance nécessaire au fonctionnement du robot.

COMMENT CHOISIR LE BON TYPE DE ROBOT ?

Le choix d'un robot dépend de plusieurs facteurs :

- 1 **La tâche à accomplir** : quelle opération doit être réalisée ?
- 2 **L'environnement** : l'espace est-il structuré, dynamique, dangereux, aquatique, aérien, etc. ?
- 3 **Le niveau d'autonomie souhaité** : le robot doit-il être téléopéré, semi-autonome ou totalement autonome ?
- 4 **Le budget** : coût d'achat, d'intégration, d'exploitation et de maintenance.
- 5 **La sécurité** : le robot interagira-t-il avec des humains ou opérera-t-il dans des zones à risque ?



À RETENIR

Il n'existe pas un robot universel, mais une grande diversité de robots adaptés à des besoins spécifiques. Bien connaître les types de robots est essentiel pour concevoir, utiliser et collaborer efficacement avec eux.

1

CHAPITRE 1

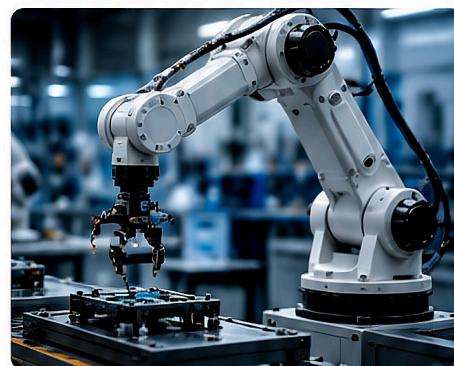
Introduction à la robotique

1.5

COMPOSANTS D'UN ROBOT

Un robot est un système complexe composé de plusieurs éléments qui travaillent ensemble pour percevoir son environnement, prendre des décisions et agir sur le monde réel.

Bien que la conception d'un robot puisse varier selon son type et son utilisation, tous les robots possèdent des composants fondamentaux. Comprendre ces composants est essentiel pour concevoir, utiliser et programmer efficacement un robot.



LES COMPOSANTS PRINCIPAUX

1. Capteurs

Perçoivent les informations de l'environnement (position, lumière, force, température, etc.).

2. Unité de contrôle

Traite les données, prend des décisions et envoie des commandes aux actionneurs.

3. Actionneurs

Produisent le mouvement ou la force nécessaire à l'exécution des tâches (moteurs, vérins, etc.).



4. Effecteur (outil)

Interagit directement avec l'environnement pour accomplir la tâche (pince, soudeuse, etc.).

5. Structure mécanique

Supporte et relie tous les composants du robot en assurant sa rigidité et sa mobilité.

6. Source d'énergie

Fournit l'énergie nécessaire au fonctionnement des composants (électricité, batterie, hydraulique, etc.).

DÉTAIL DES COMPOSANTS

1. Capteurs :

Ils collectent des données sur l'environnement interne et externe du robot.
Exemples : caméras, capteurs de distance, accéléromètres, capteurs de force.

2. Unité de contrôle :

Le « cerveau » du robot. Elle exécute le programme, analyse les données des capteurs et envoie des commandes aux actionneurs.
Exemples : microcontrôleurs, cartes électroniques, ordinateurs embarqués.

3. Actionneurs :

Ils convertissent l'énergie en mouvement ou en force.
Exemples : moteurs électriques, servomoteurs, vérins pneumatiques, moteurs pas à pas.

4. Effecteur (outil) :

La partie du robot qui interagit directement avec les objets ou l'environnement.
Exemples : pince, ventouse, outil de soudure, pulvérisateur, etc.

5. Structure mécanique :

Elle assure le support, la mobilité et la protection des composants.
Fabriquée en matériaux résistants comme l'aluminium, l'acier ou les composites.

6. Source d'énergie :

Elle alimente les composants du robot.
Exemples : alimentation électrique, batterie, système hydraulique ou pneumatique.

EXEMPLES DE RÔLE DES COMPOSANTS

Composant	Rôle principal	Exemple concret
Capteurs	Détecter et mesurer des informations	Caméra pour repérer une pièce
Unité de contrôle	Traiter les données et prendre des décisions	Carte Arduino qui lit un capteur et commande un moteur
Actionneurs	Produire un mouvement ou une force	Moteur qui fait tourner un bras robotique
Effecteur	Effectuer l'action souhaitée	Pince qui saisit et déplace un objet
Structure mécanique	Supporter et lier tous les éléments	Bras en aluminium qui maintient les moteurs et les axes
Source d'énergie	Fournir l'énergie nécessaire	Batterie Li-ion qui alimente le robot mobile



À RETENIR

Tous les robots, qu'ils soient simples ou avancés, reposent sur des composants essentiels qui travaillent ensemble. Bien comprendre leur rôle permet de mieux concevoir, utiliser et entretenir un robot.

1

CHAPITRE 1

Introduction à la robotique

1.6

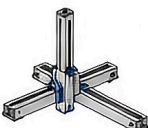
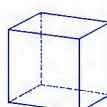
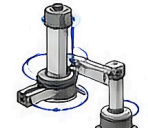
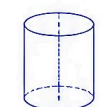
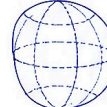
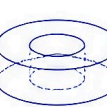
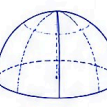
CONFIGURATIONS DES ROBOTS

La configuration d'un robot correspond à la disposition de ses axes de mouvement. Elle détermine l'espace de travail (zone atteignable), la précision, la charge utile et l'orientation possible de l'effecteur. Chaque configuration présente des avantages et des limites qui la rendent plus adaptée à certaines applications.

Comprendre les principales configurations permet de choisir le robot le mieux adapté à une tâche donnée et d'optimiser son implantation.



PRINCIPALES CONFIGURATIONS DE ROBOTS MANIPULATEURS

Configuration	Description	Axes typiques	Espace de travail	Avantages	Applications courantes
 1. Cartésienne (PPP)	Mouvements linéaires selon trois axes orthogonaux (X, Y, Z). Le poignet est fixe.	3 axes linéaires (X, Y, Z)	 Parallélépipédique	- Très grande précision - Structure simple - Facile à programmer	- Assemblage - Machines-outils - Impression 3D - Manipulation de pièces
 2. Cylindrique (RPP)	Rotation autour d'une base (R), suivie de deux mouvements linéaires (P, P).	1 rotation + 2 linéaires (θ , Z, R)	 Cylindrique	- Grande capacité d'extension - Bonne accessibilité - Structure compacte	- Manipulation - Chargement/déchargement - Soudage - Assemblage
 3. Sphérique / Polaire (RRP)	Rotation de la base (R), rotation du bras (R), puis mouvement linéaire (P).	2 rotations + 1 linéaire (θ , ϕ , r)	 Sphérique	- Grand espace de travail - Bonne portée - Structure robuste	- Soudage à l'arc - Manutention lourde - Fonderie - Peinture
 4. Articulée (RRR ou plus)	Plusieurs articulations rotatives en série permettant une grande liberté de mouvement.	3 à 6 rotations (selon le modèle)	 Sphérique (irrégulier)	- Très grande flexibilité - Orientation de l'outil dans toutes les directions	- Soudage - Peinture - Assemblage complexe - Usinage
 5. SCARA (RRP ou RRP)	Deux rotations horizontales (R, R) pour le positionnement dans le plan, puis un mouvement linéaire vertical (P).	2 rotations + 1 linéaire (θ_1 , θ_2 , Z)	 Cylindrique (en couronne)	- Rapide et précis - Idéal pour les tâches d'assemblage rapides	- Assemblage électronique - Pick and place - Conditionnement - Visage, vissage
 6. Delta (Parallèle)	Bras légers montés en parallèle permettant un mouvement très rapide de l'effecteur.	3 rotations parallèles	 Hémisphérique	- Très haute vitesse - Excellente précision - Structure légère	- Pick and place ultra-rapide - Emballage - Tri et manipulation de petits objets

CHOISIR LA BONNE CONFIGURATION

Le choix dépend de plusieurs critères :

- L'espace de travail requis (forme, volume, accessibilité)
- La précision nécessaire
- La charge utile et la portée
- La vitesse d'exécution
- L'environnement et les contraintes d'installation

Il n'existe pas de configuration universelle : chaque application a son robot idéal.

ESPACE DE TRAVAIL

L'espace de travail est la zone dans laquelle l'effecteur peut se déplacer.

Sa forme varie selon la configuration du robot (parallélépipédique, cylindrique, sphérique, etc.).

Il est important de s'assurer que cette zone couvre toutes les positions nécessaires pour accomplir la tâche.



À RETENIR

La configuration d'un robot influence directement ses capacités et ses performances.

Bien comprendre les différentes configurations permet de concevoir et d'utiliser des systèmes robotisés efficaces et adaptés.

1

CHAPITRE 1

Introduction à la robotique

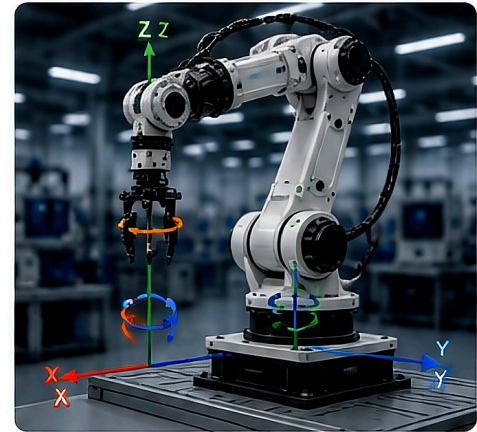
1.7

DEGRÉS DE LIBERTÉ (DDL)

Le degré de liberté (DDL) d'un robot correspond au nombre de mouvements indépendants qu'il peut effectuer pour positionner et orienter son effecteur dans l'espace.

Plus un robot possède de degrés de liberté, plus il est capable d'atteindre des positions complexes et d'orienter son outil dans différentes directions.

Le nombre de DDL influence la flexibilité du robot, mais aussi sa complexité, son coût et la difficulté de programmation.



NOMBRE DE DDL SELON LA TÂCHE

DDL	Capacités du robot	Exemples d'utilisation	Remarques
1 à 2 DDL	Mouvements simples (translation ou rotation sur un seul axe).	- Déplacement vertical d'une pièce - Orientation simple - Petites machines d'assemblage	Adapté aux tâches répétitives et peu complexes.
3 DDL	Positionnement dans l'espace (en X, Y, Z).	- Pick and place - Impression 3D - Soudage simple	Permet d'atteindre n'importe quel point de l'espace.
4 à 5 DDL	Positionnement + orientation partielle de l'effecteur.	- Soudage avec orientation - Assemblage avec inclinaison - Application d'adhésif	Bonne flexibilité pour de nombreuses applications.
6 DDL et plus	Positionnement complet et orientation libre dans toutes les directions.	- Usinage 5 axes - Soudage complexe - Manipulation dans des espaces restreints	Offre une grande liberté de mouvement mais augmente la complexité et le coût.

TYPES DE MOUVEMENTS ÉLÉMENTAIRES

- Translation linéaire** : déplacement en ligne droite selon un axe (X, Y ou Z).
- Rotation** : mouvement de pivotement autour d'un axe.
- Combinaison** : la plupart des robots combinent des translations et des rotations pour atteindre et orienter l'effecteur.

Un robot manipulateur industriel classique à 6 DDL possède généralement la structure suivante :

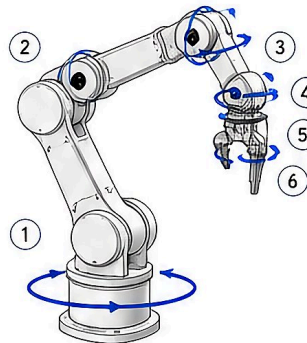
- Base** : rotation autour de l'axe vertical (axe 1).
- Épaule** : rotation qui lève ou abaisse le bras (axe 2).
- Coude** : rotation qui plie le bras (axe 3).
- Poignet 1** : rotation pour orienter l'outil (axe 4).
- Poignet 2** : inclinaison de l'outil (axe 5).
- Poignet 3** : rotation de l'outil (axe 6).



REMARQUE

Le nombre de DDL n'est pas toujours égal au nombre d'axes mécaniques. Certaines configurations utilisent des axes couplés ou des liaisons parallèles.

EXEMPLE : ROBOT 6 DDL



- Axe 1 (Base) : rotation gauche / droite
- Axe 2 (Épaule) : haut / bas
- Axe 3 (Coude) : plier / étendre
- Axe 4 (Poignet 1) : rotation
- Axe 5 (Poignet 2) : inclinaison
- Axe 6 (Poignet 3) : rotation de l'outil

Grâce à ses 6 DDL, le robot peut atteindre n'importe quel point de son espace de travail et orienter son outil selon n'importe quelle direction.

COMMENT CHOISIR LE NOMBRE DE DDL ?

Le choix dépend des besoins de l'application :

- Simple transfert d'objets → 3 DDL peuvent suffire.
- Assemblage ou soudage complexe → 4 à 6 DDL recommandés.
- Usinage 5 axes ou manipulation en espaces restreints → 6 DDL ou plus.
- Plus de DDL = plus de flexibilité, mais aussi plus de coût et de complexité.

Tâche	DDL recommandés	Exemples
Transfert simple	2 à 3	Pick and place, triage
Assemblage léger	3 à 4	Vissage, collage
Soudage / Peinture	4 à 6	Soudage à l'arc, peinture robotisée
Usinage 5 axes	6 ou plus	Fraisage, découpe complexe



À RETENIR

Le degré de liberté est un critère essentiel pour évaluer les capacités d'un robot. Choisir le bon nombre de DDL permet d'obtenir la performance nécessaire sans surdimensionner le système.

1

CHAPITRE 1 Introduction à la robotique

1.8

ESPACE DE TRAVAIL ET ENVELOPPE OPÉRATIONNELLE

L'espace de travail d'un robot correspond à l'ensemble des positions que son effecteur peut atteindre dans l'espace.

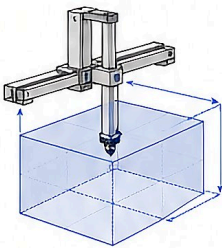
Il est défini par la cinématique du robot, ses limitations mécaniques et les contraintes de l'environnement.

Connaître l'espace de travail est essentiel pour vérifier si le robot peut atteindre toutes les points nécessaires à une tâche et pour planifier son implantation.



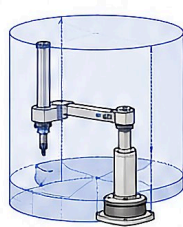
FORMES D'ESPACE DE TRAVAIL SELON LA CONFIGURATION

**Cartésienne
(PPP)**



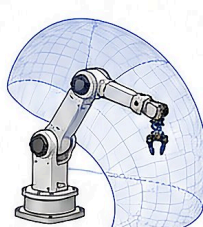
Enveloppe parallélépipédique (forme de boîte).
Adaptée aux mouvements linéaires et aux tâches de grande précision.

**Cylindrique
(RPP)**



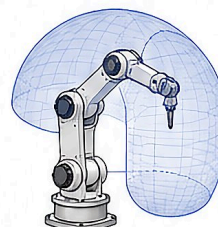
Enveloppe cylindrique.
Grande capacité d'accès autour d'un axe vertical.
Idéale pour le chargement et déchargement.

**Sphérique / Polaire
(RRP)**



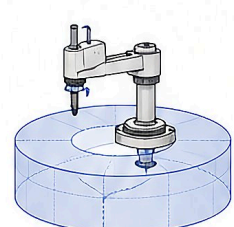
Enveloppe sphérique (ou portion de sphère).
Bonne portée et grande flexibilité d'orientation.

**Articulée
(RRR ou plus)**



Enveloppe complexe et irrégulière.
Très grande flexibilité dans toutes les directions.

**SCARA
(RRP ou RRP)**



Enveloppe en forme de cylindre évidé.
Idéale pour l'assemblage rapide de pièces plates.

FACTEURS QUI INFLUENCENT L'ESPACE DE TRAVAIL

- **Limites mécaniques** : butées des articulations et courses des axes.
- **Dimensions des liens** : longueur des bras et déport de l'outil.
- **Charge utile** : une charge plus élevée peut réduire l'espace de travail sûr.
- **Obstacles et environnement** : présence de machines, murs, zones de sécurité.
- **Outil ou effecteur** : sa taille peut réduire l'accès à certaines zones.

CALCUL DE L'ESPACE DE TRAVAIL

Le calcul exact de l'espace de travail dépend du modèle cinématique du robot.

Il peut être déterminé par :

- Des méthodes géométriques (cinématique directe et inverse)
- Des simulations numériques (logiciels de CAO/robotique)
- Des mesures expérimentales

Ces méthodes permettent de visualiser et de vérifier l'atteignabilité des points dans l'espace.

EXEMPLES D'APPLICATION

Application	Exigence d'espace de travail	Configuration recommandée	Remarques
Soudage d'une grande structure	Grande portée et accès 3D	Articulée (6 DDL)	Nécessite une grande flexibilité d'orientation.
Pick and place sur ligne de production	Accès rapide en zone plane	SCARA	Mouvements rapides et répétitifs sur un plan horizontal.
Assemblage de pièces volumineuses	Grand volume de travail	Cylindrique ou Articulée	Accès autour de la pièce.
Usinage ou impression 3D	Précision linéaire sur 3 axes	Cartésienne	Excellente précision et rigidité.
Soudage à l'arc complexe	Orientation dans toutes les directions	Articulée (6 DDL)	Accès aux zones difficiles.



À RETENIR

L'espace de travail définit les limites physiques d'action d'un robot.
Bien le comprendre permet d'assurer que le robot peut accomplir la tâche prévue, tout en optimisant son implantation, sa sécurité et son efficacité.

1

CHAPITRE 1

Introduction à la robotique

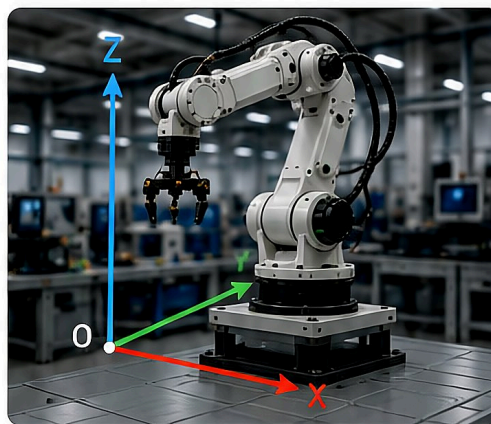
1.9

SYSTÈMES DE COORDONNÉES ET REPÈRES

Pour contrôler la position et l'orientation d'un robot, il est essentiel de décrire les mouvements dans un système de coordonnées.

Les systèmes de coordonnées permettent de représenter les positions des points et les mouvements dans l'espace de manière mathématique.

Ils sont utilisés en programmation, en simulation et dans la commande réelle du robot.



TYPES DE SYSTÈMES DE COORDONNÉES

Système	Description	Axes	Utilisation	Illustration
Cartésien (rectangulaire)	Basé sur trois axes linéaires perpendiculaires entre eux. La position d'un point est définie par (X, Y, Z).	X, Y, Z orthogonaux	Le plus courant en robotique. Utilisé pour décrire la position de l'effecteur dans l'espace.	
Cylindrique	Combinaison d'une rotation autour de l'axe Z et de deux déplacements linéaires. La position est définie par (ρ, θ, Z).	ρ (rayon), θ (angle), Z (hauteur)	Utilisé pour les robots cylindriques et pour les problèmes à symétrie axiale.	
Sphérique (polaire)	Basé sur la distance à l'origine et deux angles. La position est définie par (r, θ, φ).	r (rayon), θ (azimut), φ (élévation)	Utilisé pour les robots sphériques et pour les calculs impliquant des surfaces sphériques.	
Articulaire (angulaire)	Les positions sont définies par les angles des articulations du robot. La position est définie par (θ ₁ , θ ₂ , ..., θ _n).	θ ₁ , θ ₂ , ..., θ _n (angles des articulations)	Utilisé pour décrire la configuration interne du robot et en cinématique inverse.	

REPÈRES UTILISÉS EN ROBOTIQUE

- **Repère de base (ou repère monde) :**
Fixe dans l'environnement. Toutes les positions du robot sont exprimées par rapport à ce repère.
- **Repère de l'outil (ou repère effecteur) :**
Attaché à l'outil du robot. Il définit l'orientation et la position de l'outil par rapport à la base.
- **Repères intermédiaires :**
Attachés à chaque lien ou articulation du robot. Ils facilitent le calcul des transformations.

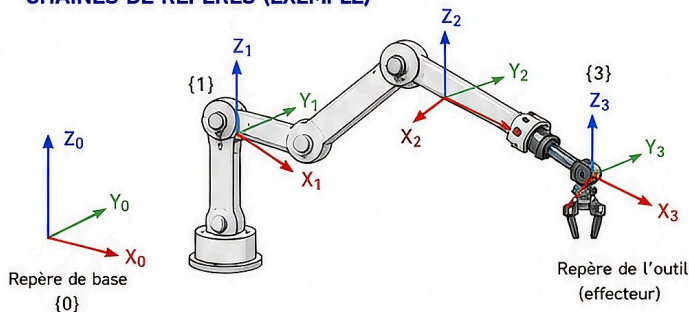
TRANSFORMATION ENTRE REPÈRES

Pour passer d'un repère à un autre, on utilise des transformations homogènes représentées par des matrices 4x4.

Ces matrices combinent rotation et translation.

$${}^A_T B = \begin{bmatrix} R_{3 \times 3} & P_{3 \times 1} \\ 0_{1 \times 3} & 1 \end{bmatrix} \quad \begin{array}{l} R : \text{matrice de rotation} \\ P : \text{vecteur de position} \\ 0 : \text{vecteur ligne nul} \end{array}$$

CHAINES DE REPÈRES (EXEMPLE)



POURQUOI SONT-ILS IMPORTANTS ?

- Permettent de définir précisément la position et l'orientation du robot.
- Indispensables pour la cinématique directe et inverse.
- Essentiels pour la programmation hors ligne (simulation) et en ligne (contrôle).
- Facilitent la communication entre le robot et les systèmes externes (capteurs, caméras, machines).



À RETENIR

Les systèmes de coordonnées et les repères permettent de décrire les mouvements du robot dans l'espace. Une bonne compréhension est essentielle pour programmer, simuler et contrôler un robot efficacement.

1

CHAPITRE 1

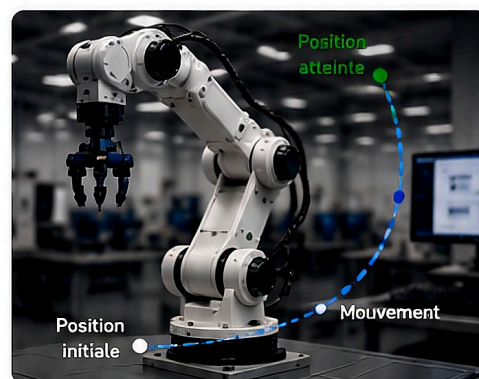
Introduction à la robotique

1.10

CINÉMATIQUE DIRECTE ET CINÉMATIQUE INVERSE

La cinématique étudie le mouvement des robots sans tenir compte des forces qui les produisent. Elle se divise principalement en deux problèmes complémentaires : la cinématique directe et la cinématique inverse.

Comprendre ces deux notions est essentiel pour modéliser, simuler et contrôler un robot efficacement.



CINÉMATIQUE DIRECTE (CD)

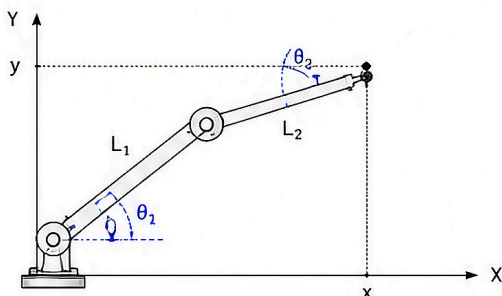
La cinématique directe consiste à déterminer la position et l'orientation de l'organe terminal (effecteur) à partir des paramètres articulaires connus (angles ou déplacements des axes).

En résumé :

CD : paramètres articulaires → position et orientation de l'effecteur

Exemple :

Pour un bras robotique à 2 axes (plan), si l'on connaît les angles θ_1 et θ_2 des deux articulations, on peut calculer la position (x, y) de l'effecteur.



Équations (exemple 2 axes plan) :

$$\begin{aligned} x &= L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) \\ y &= L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) \end{aligned}$$

CINÉMATIQUE INVERSE (CI)

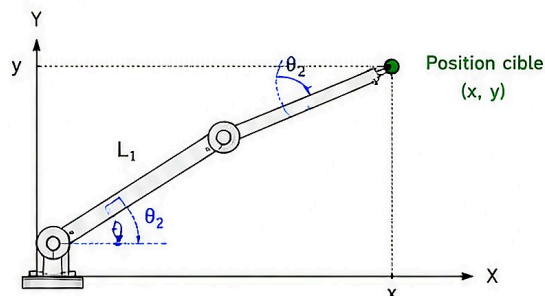
La cinématique inverse consiste à déterminer les paramètres articulaires nécessaires pour atteindre une position et une orientation souhaitées de l'effecteur.

En résumé :

CI : position et orientation de l'effecteur → paramètres articulaires

Exemple :

Étant donnée une position cible (x, y) , on cherche les angles θ_1 et θ_2 qui permettent d'y amener l'effecteur.



Remarque :

La cinématique inverse peut avoir zéro, une ou plusieurs solutions selon la position cible et la configuration du robot.

COMPARAISON ENTRE CD ET CI

Aspect	Cinématique directe (CD)	Cinématique inverse (CI)
Entrée	Paramètres articulaires (angles, déplacements)	Position et orientation désirées de l'effecteur
Sortie	Position et orientation de l'effecteur	Paramètres articulaires nécessaires
Complexité	Généralement simple à calculer	Peut être complexe, plusieurs solutions possibles
Utilisation principale	Simulation, visualisation, vérification de mouvement	Planification de trajectoire, commande du robot
Solutions multiples	Non	Oui, parfois plusieurs configurations possibles
Exemple d'application	Où se trouve l'outil si les angles sont θ_1 et θ_2 ?	Quels angles pour atteindre le point (x, y, z) ?

MÉTHODES DE RÉOLUTION

- Cinématique directe :**
 - Utilise les matrices de transformation homogènes (méthode de Denavit–Hartenberg) ou les équations géométriques.
- Cinématique inverse :**
 - Solutions analytiques pour les robots simples.
 - Méthodes numériques (itératives) pour les robots complexes.
 - Algorithmes d'optimisation (ex. : descente de gradient).

POINTS CLÉS

- La cinématique directe répond à "où va le robot ?"
- La cinématique inverse répond à "comment y aller ?"
- La maîtrise de ces deux notions est indispensable pour le contrôle et la planification des mouvements robotiques.



À RETENIR

La cinématique directe et inverse forment la base mathématique du mouvement des robots. Elles permettent de passer des angles aux positions et inversement, pour déplacer l'effecteur avec précision.

1

CHAPITRE 1

Introduction à la robotique

1.11

MODÉLISATION CINÉMATIQUE D'UN ROBOT

La modélisation cinématique consiste à représenter mathématiquement la géométrie d'un robot afin de décrire ses mouvements.

Un modèle cinématique permet de prédire la position et l'orientation de l'effecteur final en fonction des paramètres articulaires, ou inversement.

Ce modèle est la base de la simulation, de la planification de trajectoires et du contrôle du robot.



ÉLÉMENTS D'UN MODÈLE CINÉMATIQUE

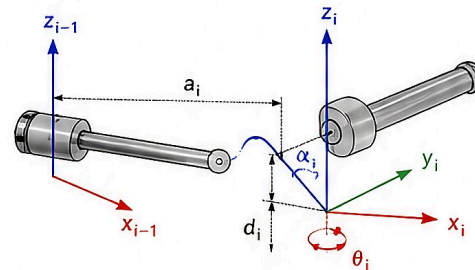
Élément	Description	Symbole / Exemple	Rôle dans le modèle
Lien	Corps rigide entre deux articulations consécutives.	 Lien i	Définit la géométrie du robot.
Articulation	Permet le mouvement relatif entre deux liens.	 Révolute (R) Prismatique (P)	Définit les degrés de liberté du robot.
Repère	Système de coordonnées attaché à chaque lien ou articulation.		Permet de décrire la position et l'orientation des liens.
Paramètres cinématiques	Ensemble des paramètres (angles, distances) qui décrivent la configuration du robot.	θ_i (rotation) d_i (translation)	Entrées du modèle cinématique.
Effecteur final	Organe terminal du robot qui interagit avec l'environnement.		Sortie du modèle : position et orientation dans l'espace.

PARAMÈTRES DE DENAVIT–HARTENBERG (DH)

La méthode de Denavit–Hartenberg est une convention standard pour décrire la géométrie d'un robot en utilisant quatre paramètres par articulation.

Paramètre	Symbole	Description	Unité
Angle de rotation	θ_i	Rotation autour de l'axe z_{i-1} pour aligner x_{i-1} avec x_i .	degré (°) ou rad
Distance de translation	d_i	Translation le long de l'axe z_{i-1} pour atteindre x_i .	m
Longueur de lien	a_i	Distance le long de l'axe x_i entre z_{i-1} et z_i .	m
Angle de torsion	α_i	Rotation autour de l'axe x_i pour aligner z_{i-1} avec z_i .	degré (°) ou rad

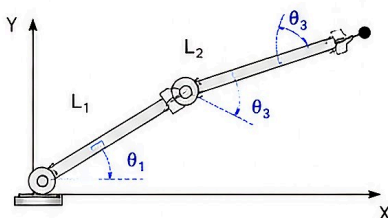
CONVENTION DE DENAVIT–HARTENBERG



θ_i : angle de rotation (autour de z_{i-1})
 d_i : distance de translation (le long de z_{i-1})
 a_i : longueur de lien (le long de x_i)
 α_i : angle de torsion (autour de x_i)

EXEMPLE DE CHAÎNE CINÉMATIQUE (3R PLAN)

Robot à 3 articulations rotatives dans un plan (X–Y).



Position de l'effecteur :

$$\begin{aligned} x &= L_1 \cos \theta_1 + L_2 \cos (\theta_1 + \theta_2) \\ &\quad + L_3 \cos (\theta_1 + \theta_2 + \theta_3) \\ y &= L_1 \sin \theta_1 + L_2 \sin (\theta_1 + \theta_2) \\ &\quad + L_3 \sin (\theta_1 + \theta_2 + \theta_3) \end{aligned}$$

Orientation :

$$\phi = \theta_1 + \theta_2 + \theta_3$$

APPLICATIONS DE LA MODÉLISATION CINÉMATIQUE

- Simulation et visualisation des mouvements du robot.
- Planification de trajectoires (éviter les collisions, optimiser le temps).
- Commande du robot en position ou en vitesse.
- Analyse des singularités et des limites de mouvement.
- Conception et dimensionnement de nouveaux robots.



À RETENIR

La modélisation cinématique est essentielle pour comprendre et prédire le comportement d'un robot. Elle repose sur la géométrie du robot, les systèmes de coordonnées et les paramètres articulaires. Maîtriser ces concepts est la première étape vers la programmation et le contrôle avancé des robots.

1

CHAPITRE 1

Introduction à la robotique

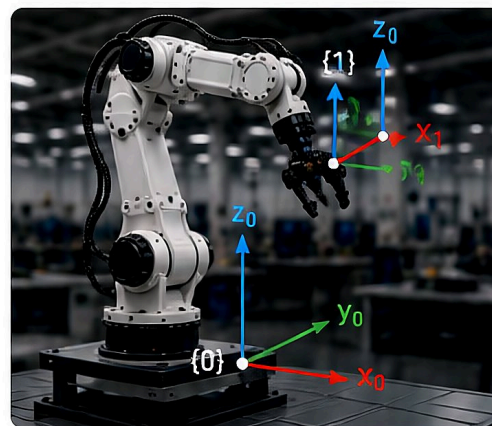
1.12

TRANSFORMATIONS HOMOGÈNES

Les transformations homogènes sont un outil mathématique puissant qui permettent de représenter la position et l'orientation d'un repère par rapport à un autre repère dans l'espace.

Elles combinent rotation et translation dans une seule matrice 4x4, ce qui simplifie les calculs en cinématique des robots.

Les matrices homogènes sont la base de la méthode de Denavit–Hartenberg et de nombreuses applications en robotique.



MATRICE DE TRANSFORMATION HOMOGÈNE

La transformation homogène qui décrit la position et l'orientation d'un repère {1} par rapport à un repère {0} est donnée par une matrice 4x4 :

$${}^0T_1 = \begin{bmatrix} R_{3 \times 3} & p_{3 \times 1} \\ \mathbf{0}_{1 \times 3} & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

où :

$R_{3 \times 3}$: matrice de rotation (orientation de {1} par rapport à {0})

$p_{3 \times 1}$: vecteur position (origine de {1} exprimée dans {0})

$\mathbf{0}_{1 \times 3}$: vecteur ligne nul

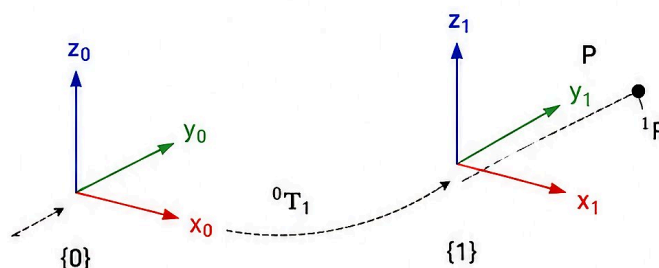
1 : élément scalaire

INTERPRÉTATION GÉOMÉTRIQUE

La matrice 0T_1 permet de transformer un point exprimé dans {1} en ses coordonnées équivalentes dans {0} :

$${}^0P = {}^0T_1 {}^1P$$

avec ${}^1P = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$ et ${}^0P = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$



PROPRIÉTÉS IMPORTANTES

• Inverse :

$${}^1T_0 = ({}^0T_1)^{-1} = \begin{bmatrix} R^T & -R^T p \\ \mathbf{0} & 1 \end{bmatrix}$$

• Produit (chaînage) :

$${}^0T_2 = {}^0T_1 {}^1T_2$$

• Déterminant :

$$\det({}^0T_1) = 1$$

• Matrice orthogonale (partie rotation) :

$$R^T R = I$$

OPÉRATIONS SUR LES TRANSFORMATIONS

- **Composition** : permet de combiner plusieurs mouvements.
- **Inversion** : permet de revenir d'un repère fils vers son parent.
- **Application à un point** : transforme les coordonnées d'un point d'un repère à un autre.

Exemple de compositions :

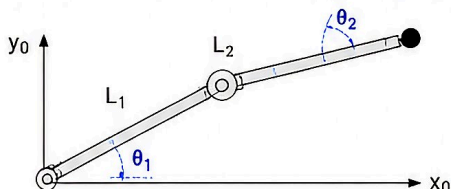
$${}^0T_3 = {}^0T_1 {}^1T_2 {}^2T_3$$

transforme un point exprimé dans {3}

en ses coordonnées dans {0} à travers les repères intermédiaires {1} et {2}.

EXEMPLE : ROBOT PLAN 2 AXES

Considérons un robot plan 2R avec des longueurs L_1 et L_2 et des angles θ_1 et θ_2 .



La transformation de {0} à {2} est :

$${}^0T_2 = \begin{bmatrix} \cos(\theta_1 + \theta_2) & -\sin(\theta_1 + \theta_2) & 0 & L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) & \cos(\theta_1 + \theta_2) & 0 & L_1 \sin \theta_1 + L_2 \sin(\theta_1 + \theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

APPLICATIONS EN ROBOTIQUE

- Modélisation cinématique (directe et inverse)
- Planification de trajectoires
- Simulation et animation de robots
- Vision robotique et localisation
- Contrôle de mouvement



À RETENIR

Les transformations homogènes unifient rotation et translation dans une seule matrice. Elles simplifient les calculs et sont essentielles pour modéliser les chaînes cinématiques des robots. Maîtriser cet outil est indispensable pour comprendre la robotique moderne.

1

CHAPITRE 1

Introduction à la robotique



1.13

ACTIONNEURS ET MOTEURS

Les muscles du robot

Les actionneurs sont les composants qui convertissent l'énergie (électrique, pneumatique, hydraulique, etc.) en mouvement mécanique pour produire l'action désirée par le robot.

Ils sont essentiels pour déplacer les articulations, les effecteurs finaux et les systèmes mobiles.



TYPES D'ACTIONNEURS COURAMMENT UTILISÉS



Moteur à courant continu (DC)



- Simple et économique
- Vitesse variable
- Utilisé dans les roues, les bras légers, etc.



Moteur pas à pas (Stepper)



- Mouvement précis par pas
- Excellent contrôle de position
- Imprimantes 3D, bras robotique, CNC, etc.



Servomoteur



- Contrôle de position intégré
- Couple élevé dans un format compact
- Idéal pour les articulations et les effecteurs



Actionneur linéaire (électrique)



- Produit un mouvement linéaire
- Utilisé pour pousser, tirer ou lever



Actionneur pneumatique ou hydraulique



- Grande force et vitesse
- Utilisé dans les robots industriels lourds
- Nécessite un compresseur (pneumatique)

CRITÈRES DE CHOIX D'UN ACTIONNEUR



Couple / Force : Doit être suffisant pour vaincre les charges et les frottements.



Vitesse : Selon l'application (rapide ou précise).



Précision : Capacité à atteindre et maintenir une position.



Taille et poids : Doivent s'adapter à la conception du robot.



Consommation énergétique : Impacte l'autonomie du robot.



Type d'alimentation : Électrique, pneumatique, hydraulique.

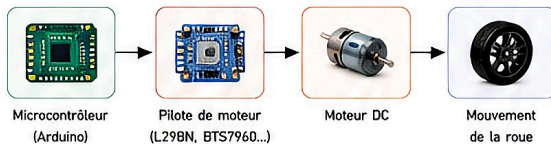


Coût : Important selon le budget du projet.

COMPARAISON RAPIDE

Type	Couple	Précision	Vitesse	Complexité	Coût
Moteur DC	Moyen ★★★☆☆	Moyenne ☆☆☆☆☆	Élevée ★★★★★	Faible ★★★★☆	\$\$\$
Moteur pas à pas	Moyen ★★★★★	Élevée ☆☆☆☆☆	Moyenne ★★★☆☆	Moyenne ★★★★☆	\$\$
Servomoteur	Élevé ★★★★★	Élevée ☆☆☆☆☆	Élevée ★★★★★	Faible ★★☆☆☆	Moyen à élevé \$\$\$
Actionneur linéaire	Élevé ★★★★★	Élevée ☆☆☆☆☆	Moyenne ★★★☆☆	Moyenne ★★★★☆	\$\$\$
Pneumatique / Hydraulique	Très élevé ★★★★★	Moyenne ☆☆☆☆☆	Élevée ★★★★★	Élevée ★★★★☆	\$\$\$

COMMENT PILOTER UN MOTEUR ? (EXEMPLE : MOTEUR DC)



Fonctions du pilote :

- Fournit le courant nécessaire au moteur.
- Permet de contrôler le sens et la vitesse (PWM).
- Protège le microcontrôleur.



ASTUCE

Utilisez la modulation PWM pour régler la vitesse et économiser l'énergie.

EXEMPLES D'APPLICATIONS



Robot mobile

Les moteurs DC font tourner les roues.



Bras robotique

Les servomoteurs contrôlent chaque articulation.



Système de levage

Les actionneurs linéaires soulèvent ou abaissent une charge.



Automatisation industrielle

Les vérins pneumatiques réalisent des mouvements rapides et puissants.



BONNES PRATIQUES

- Choisir l'actionneur adapté à la tâche à accomplir.
- Prévoir une marge de couple pour éviter les surcharges.
- Protéger les moteurs contre la surchauffe et les surtensions.
- Tester le système sous charge réelle.



À RETENIR

Les actionneurs et moteurs sont les « muscles » du robot. Un bon choix et un bon contrôle des actionneurs garantissent des mouvements précis, efficaces et fiables.

1

CHAPITRE 1

Introduction à la robotique

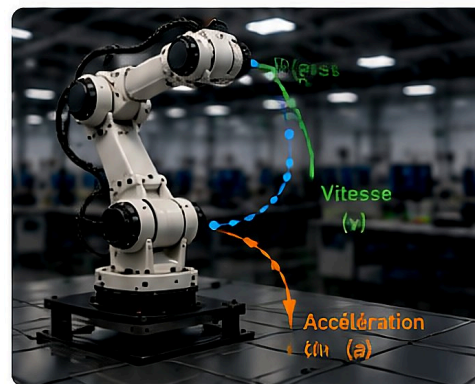
1.14

VITESSE ET ACCÉLÉRATION EN ROBOTIQUE

Pour contrôler un robot en mouvement, il ne suffit pas de connaître sa position. Il est également essentiel de connaître sa vitesse et son accélération.

Ces grandeurs permettent de planifier des trajectoires lisses, d'éviter les chocs et d'optimiser la performance du robot.

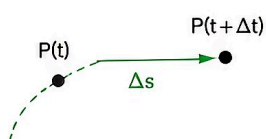
On distingue la cinématique (géométrie du mouvement) de la dynamique (forces et couples). Ici, nous nous concentrons sur la cinématique.



VITESSE LINÉAIRE ET VITESSE ANGULAIRE

Vitesse linéaire

Décrit la vitesse à laquelle un point se déplace dans l'espace.



Formule :

$$\mathbf{v} = \lim_{\Delta t \rightarrow 0} \frac{\Delta \mathbf{r}}{\Delta t} = \frac{d\mathbf{r}}{dt}$$

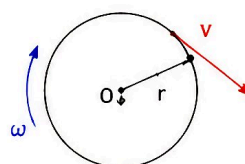
où :

$\mathbf{v}$: vitesse linéaire (m/s)

$\mathbf{r}$: vecteur position (m)

Vitesse angulaire

Décrit la vitesse de rotation autour d'un axe.



Formule :

$$\omega = \frac{d\theta}{dt}$$

où :

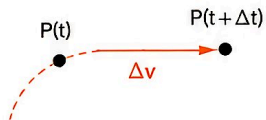
ω : vitesse angulaire (rad/s)

θ : angle (rad)

ACCÉLÉRATION LINÉAIRE ET ANGULAIRE

Accélération linéaire

Décrit la variation de la vitesse linéaire.



Formule :

$$\mathbf{a} = \lim_{\Delta t \rightarrow 0} \frac{\Delta \mathbf{v}}{\Delta t} = \frac{d\mathbf{v}}{dt}$$

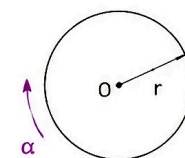
où :

$\mathbf{a}$: accélération linéaire (m/s²)

$\mathbf{v}$: vitesse linéaire (m/s)

Accélération angulaire

Décrit la variation de la vitesse angulaire.



Formule :

$$\alpha = \frac{d\omega}{dt}$$

où :

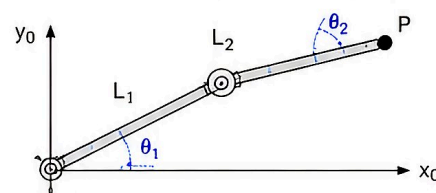
α : accélération angulaire (rad/s²)

ω : vitesse angulaire (rad/s)

RELATIONS ENTRE GRANDEURS

Relation	Description	Formule
Lien vitesse linéaire – angulaire	Pour un point à distance r de l'axe de rotation	$\mathbf{v} = \omega \times \mathbf{r}$ ($ \mathbf{v} = \omega r$ si perpendiculaire)
Lien accélération linéaire – angulaire	Pour un point à distance r de l'axe de rotation	$\mathbf{a} = \alpha \times \mathbf{r} + \omega \times (\omega \times \mathbf{r})$
Accélération tangentielle	Composante due à l'accélération angulaire	$a_t = \alpha r$
Accélération centripète	Composante due à la rotation	$a_c = \omega^2 r$ (dirigée vers l'axe)

EXEMPLE : BRAS ROBOTIQUE PLAN 2R



La vitesse linéaire de l'effecteur P est :

$$\mathbf{v}_P = \mathbf{J}(\theta) \dot{\theta}$$

où :

$\mathbf{J}(\theta)$: matrice jacobienne (2×2)

$$\dot{\theta} = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} : \text{vecteur des vitesses articulaires}$$

MATRICE JACOBIEENNE

La jacobienne relie les vitesses articulaires aux vitesses linéaires de l'effecteur : $\mathbf{v} = \mathbf{J}(\theta) \dot{\theta}$

Pour un robot plan 2R :

$$\mathbf{J}(\theta) = \begin{bmatrix} -L_1 \sin \theta_1 - L_2 \sin(\theta_1 + \theta_2) & -L_2 \sin(\theta_1 + \theta_2) \\ L_1 \cos \theta_1 + L_2 \cos(\theta_1 + \theta_2) & L_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

Elle permet aussi de relier les accélérations : $\mathbf{a} = \mathbf{J}(\theta) \ddot{\theta} + \dot{\mathbf{J}}(\theta) \dot{\theta}$

POURQUOI C'EST IMPORTANT ?



Permet de planifier des trajectoires lisses et rapides.



Aide à éviter les chocs mécaniques et les vibrations.



Optimise la performance et la précision du robot.



Essentiel pour le contrôle en temps réel et la sécurité.



À RETENIR

La vitesse et l'accélération décrivent comment un robot se déplace dans le temps.

Elles sont indispensables pour un contrôle précis, sûr et efficace.

La jacobienne est l'outil clé pour relier le mouvement articulaire au mouvement de l'effecteur.

1

CHAPITRE 1

Introduction à la robotique

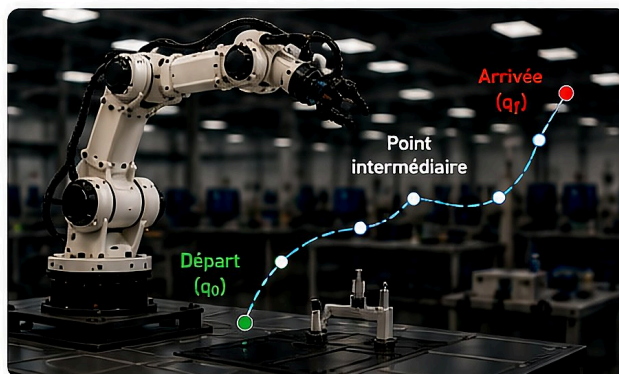
1.15

PLANIFICATION DE TRAJECTOIRES

La planification de trajectoires consiste à déterminer comment un robot doit se déplacer d'un point initial à un point final tout en satisfaisant des contraintes de mouvement et en évitant les obstacles.

Une bonne trajectoire assure un mouvement fluide, rapide, précis et sûr.

Elle est composée de la géométrie du chemin (où aller) et de la loi temporelle (à quelle vitesse).



COMPOSANTES D'UNE TRAJECTOIRE

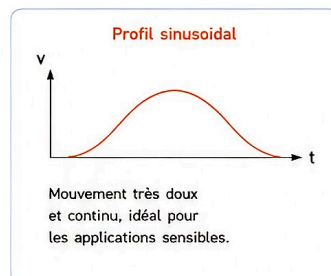
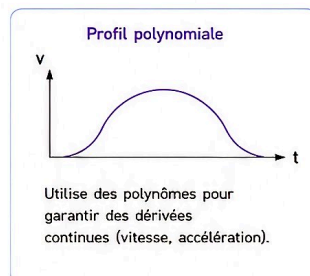
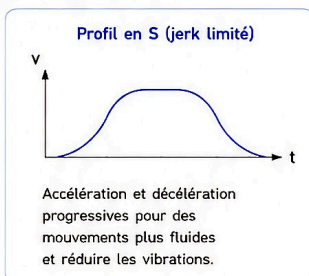
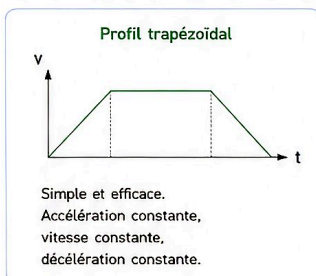
Composante	Description
Géométrie du chemin (Où ?)	Définit le chemin dans l'espace cartésien ou dans l'espace articulaire que le robot doit suivre.
Loi temporelle (Quand ?)	Spécifie l'évolution du temps le long du chemin : vitesse, accélération et durée.
Contraintes	Limites de vitesse, d'accélération, de couple, évitement d'obstacles, limites de joint.

TYPES DE TRAJECTOIRES

Point à point	Linéaire	Circulaire	Par points intermédiaires
Se déplace du point initial au point final sans imposer de chemin particulier.	Suit une ligne droite entre deux points dans l'espace cartésien.	Suit un arc ou un cercle défini par son centre et son rayon.	Passe par plusieurs points définis pour contrôler plus finement le chemin.

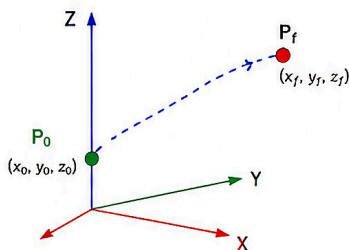
PARAMÉTRISATION TEMPORELLE

La paramétrisation temporelle associe à chaque point du chemin un temps t , afin de contrôler la vitesse et l'accélération du robot. Les profils les plus utilisés sont :



EXEMPLE : TRAJECTOIRE LINÉAIRE

Déplacer l'effecteur de $P_0 (x_0, y_0, z_0)$ à $P_f (x_f, y_f, z_f)$ en temps T avec un profil trapézoïdal.



Équation du chemin (linéaire) :

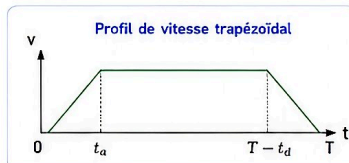
$$P(s) = P_0 + s(P_f - P_0)$$

avec $0 \leq s \leq 1$

Loi temporelle (profil trapézoïdal) :

- Accélération : $0 \leq t \leq t_a$
- Vitesse constante : $t_a < t \leq T - t_d$
- Décélération : $T - t_d < t \leq T$

où t_a et t_d sont les temps d'accélération et de décélération.



MÉTHODES DE PLANIFICATION DE TRAJECTOIRES



Méthodes géométriques

Trajectoires définies par des formes simples (droite, cercle, spline). Faciles à implémenter et rapides.



Méthodes basées sur les points de passage

Utilisent des points intermédiaires pour contrôler le chemin.



Méthodes d'optimisation

Minimisent un coût (temps, énergie, jerk) tout en respectant les contraintes.



Méthodes sampling-based (RRT, PRM)

Explorent l'espace pour trouver un chemin libre d'obstacles. Utilisées pour les environnements complexes.

CONSIDÉRATIONS PRATIQUES

- ✓ Respecter les limites de vitesse, d'accélération et de couple.
- ✓ Éviter les singularités et les collisions.
- ✓ Assurer la continuité de la trajectoire (position, vitesse, accélération).
- ✓ Adapter la trajectoire au type d'application (rapidité, précision, sécurité).
- ✓ Valider la trajectoire par simulation avant exécution.



À RETENIR

La planification de trajectoires est essentielle pour générer des mouvements sûrs, fluides et efficaces. Elle combine la géométrie du chemin, la loi temporelle et le respect des contraintes du robot et de son environnement. Maîtriser ces concepts est indispensable pour le contrôle avancé et l'optimisation des performances robotiques.

1

CHAPITRE 1

Introduction à la robotique

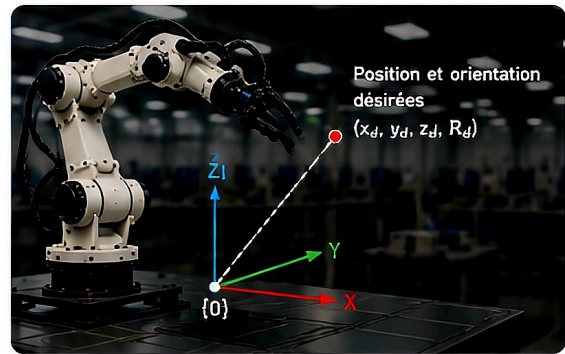
1.16

CINÉMATIQUE INVERSE

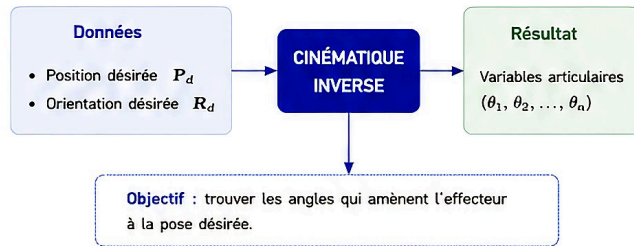
La cinématique inverse consiste à déterminer les valeurs des variables articulaires (angles ou déplacements) permettant d'atteindre une position et une orientation désirées de l'effecteur.

C'est le problème inverse de la cinématique directe.

Elle est essentielle pour commander le robot en envoyant les angles à ses moteurs.

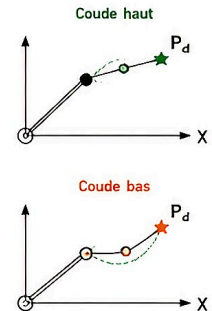


PROBLÈME DE CINÉMATIQUE INVERSE



EXISTENCE ET NOMBRE DE SOLUTIONS

- ✓ Plusieurs solutions possibles
Un même point peut être atteint avec différentes configurations (ex. : coude haut ou coude bas).
- ⚠ Aucune solution possible
Si le point désiré est en dehors de l'espace de travail du robot.
- ⚙ Solutions infinies
Pour certains robots redondants (plus de degrés de liberté que nécessaire).

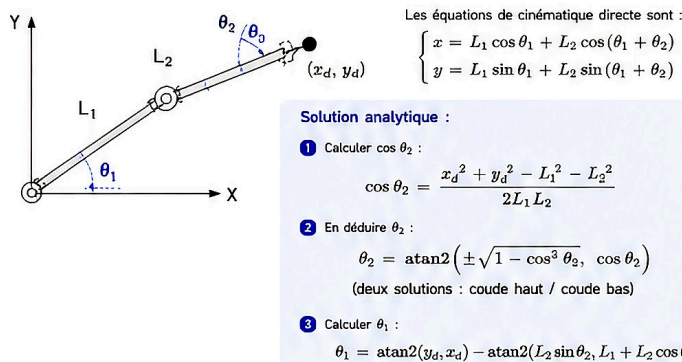


MÉTHODES POUR RÉSOUDRE LA CINÉMATIQUE INVERSE

Méthode	Description	Avantages	Inconvénients	Adaptée pour
1. Solution analytique (fermée)	Calculs mathématiques directs à partir de la géométrie du robot.	✓ Très rapide ✓ Précise	✗ Complexe à dériver ✗ Pas toujours possible (robots complexes)	Robots simples (2R, 3R, cartésien, sphérique, SCARA, etc.)
2. Méthodes itératives	Recherche numérique d'une solution en minimisant l'erreur.	✓ Applicables à presque tous les robots ✓ Gèrent les contraintes	✗ Plus lentes ✗ Dépendent d'une bonne estimation initiale	Robots complexes ou redondants
3. Approche par optimisation	Formulation comme problème d'optimisation (minimiser l'erreur).	✓ Intègre plusieurs critères (énergie, obstacles, etc.) ✓ Pour robots redondants	✗ Coût de calcul élevé	Robots redondants, planification avancée

EXEMPLE : ROBOT PLAN 2R

Considérons un robot plan 2R avec des longueurs de liens L_1 et L_2 .



PSEUDO-CODE (MÉTHODE ITÉRATIVE)

```

Entrée :  $P_d, R_d, \theta_{\text{init}}$ 
 $\theta \leftarrow \theta_{\text{init}}$ 
répéter
  Calculer pose actuelle (P, R) avec la cinématique directe
  erreur ← fonction_erreur(P, R,  $P_d, R_d$ )
  si norme(erreur) < tolérance alors
    sortir (solution trouvée)
  fin si
  Calculer le jacobien  $J(\theta)$ 
   $\theta \leftarrow \theta + \alpha J(\theta)^T \text{erreur}$  ( $\alpha$  : pas d'apprentissage)
jusqu'à max_itérations
sortie :  $\theta$  (solution approchée)
  
```

APPLICATIONS



Commande de robot

Permet d'envoyer les angles aux actionneurs pour atteindre une pose désirée.



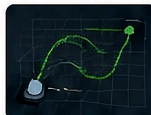
Pick and place

Atteindre précisément des positions pour saisir et déposer des objets.



Interaction humain-robot

Adapter les mouvements en temps réel selon la position de l'utilisateur.



Planification de tâches

Générer des mouvements fiables et efficaces.

BONNES PRATIQUES

- ✓ Vérifier si le point désiré est atteignable.
- ✓ Choisir la solution qui évite les obstacles et les limites de joint.
- ✓ Tenir compte des limites de vitesse et d'accélération.
- ✓ Tester la solution avec la cinématique directe.
- ✓ Pour les robots redondants, optimiser selon un critère (énergie, posture, etc.).



À RETENIR

La cinématique inverse est la clé pour convertir une pose désirée en commandes articulaires. Elle peut avoir plusieurs solutions ou aucune solution. Le choix de la méthode dépend de la complexité du robot et des exigences de l'application.

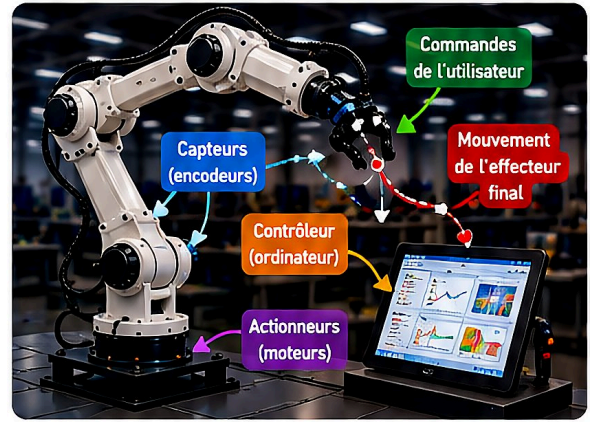
1

CHAPITRE 1 Introduction à la robotique

1.17

CONTRÔLE DES MOUVEMENTS ET COMMANDES DE ROBOTS

- Le contrôle des mouvements permet d'exécuter les trajectoires calculées en assurant précision, stabilité et sécurité du robot.
- Les commandes peuvent être en position, vitesse, effort ou couple selon les besoins de l'application.
- Un bon contrôleur réduit les erreurs et améliore les performances dynamiques du robot.



TYPES DE COMMANDES ROBOTIQUES



Commande en position

Impose une position articulaire ou cartésienne.

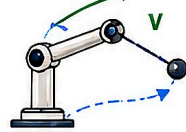


Ex. : aller à un point précis



Commande en vitesse

Impose une vitesse linéaire ou angulaire.



Ex. : vitesse constante



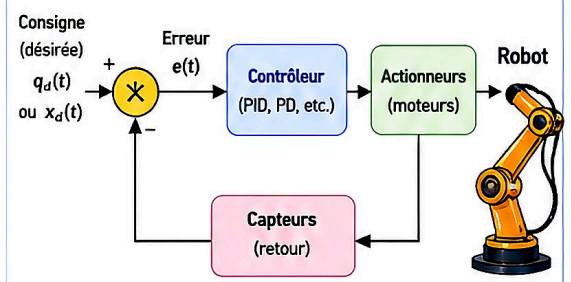
Commande en effort / couple

Impose une force ou un couple appliqué.



Ex. : serrage, polissage

STRUCTURE D'UNE BOUCLE DE CONTRÔLE



RÉGULATEUR PID (LE PLUS UTILISÉ)

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

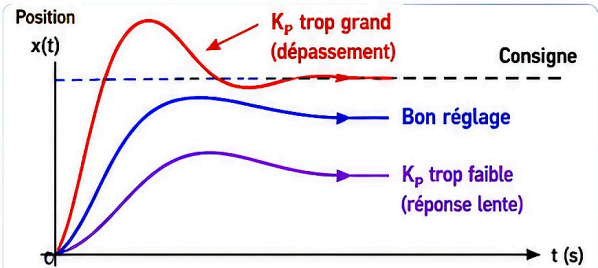
Terme proportionnel (réaction rapide) Terme intégral (élimine l'erreur statique) Terme dérivé (anticipe les variations)

Rôle des termes :

- K_p : réagit à l'erreur actuelle
- K_i : corrige l'erreur accumulée
- K_d : prévient les dépassements

Choix des gains (K_p , K_i , K_d) = réglage du contrôleur

RÉPONSE D'UN SYSTÈME À UN ÉCHELON

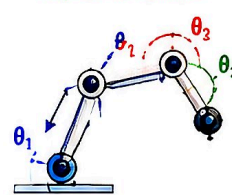


COMPARAISON DES CONTRÔLEURS

Contrôleur	Niveau	Caractéristiques	Avantages	Inconvénients
Tout ou rien 	Simple	Sortie ON/OFF	✓ Très simple ✓ Peu coûteux	✗ Oscillations ✗ Imprécis
P (Proportionnel) K_p	Débutant	$u(t) = K_p e(t)$	✓ Rapide à régler ✓ Réduction de l'erreur	✗ Erreur statique
PI K_p, K_i	Intermédiaire	$u(t) = K_p e(t) + K_i \int e(t) dt$	✓ Élimine l'erreur statique	✗ Peut overshooter
PID K_p, K_i, K_d	Avancé	$u(t) = K_p e(t) + K_i \int e(t) dt + K_d \frac{de(t)}{dt}$	✓ Précis, stable, réponse rapide	✗ Réglage délicat

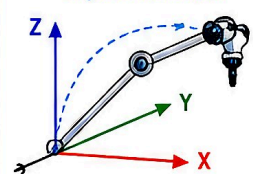
COMMANDES EN ESPACE ARTICULAIRE ET CARTÉSIEN

Espace articulaire



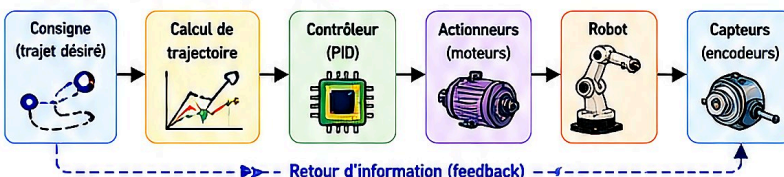
- ✓ Commande par angles (θ_i)
- ✓ Facile à implémenter
- ✓ Dépend de la cinématique

Espace cartésien



- ✓ Commande par position (x, y, z)
- ✓ Trajectoire plus intuitive
- ✓ Nécessite cinématique inverse

CHAÎNE DE CONTRÔLE COMPLÈTE



EXEMPLES D'APPLICATIONS



SÉCURITÉ ET LIMITES

- ✓ Limiter la vitesse et l'accélération.
- ✓ Détecter les collisions et arrêter le robot.
- ✓ Surveiller le couple pour protéger les actionneurs.
- ✓ Respecter les zones de travail autorisées.



SÉCURITÉ = PRIORITÉ

À RETENIR

- ✓ Le contrôle des mouvements exécute les trajectoires prévues.
- ✓ Le PID est le régulateur le plus utilisé.
- ✓ Le bon réglage des gains assure stabilité et précision.
- ✓ La sécurité et le retour d'information sont essentiels.
- ✓ Adapter le contrôle au type d'application et au robot utilisé.

1

CHAPITRE 1

Introduction à la robotique



Les capteurs sont les yeux et les sens du robot.

1.18

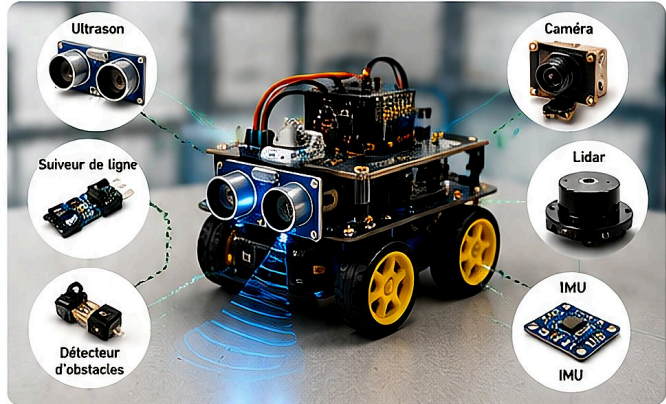
CAPTEURS EN ROBOTIQUE

Les yeux et les sens du robot

Les capteurs permettent au robot de percevoir son environnement, d'acquérir des informations et de réagir de manière appropriée.

Sans capteurs, un robot ne pourrait ni détecter d'obstacles, ni connaître sa position, ni mesurer des distances ou des forces.

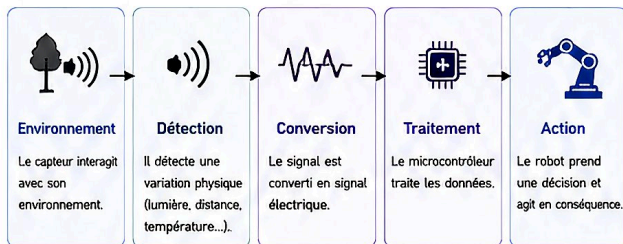
Ils sont essentiels pour réaliser des mouvements intelligents et sécurisés.



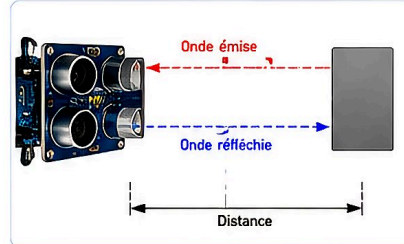
TYPES DE CAPTEURS COURAMMENT UTILISÉS

Capteurs de distance (ultrason) Mesurent la distance à un objet en émettant une onde sonore. Ex. : HC-SR04	Capteurs de distance (LiDAR) Utilisent un laser pour scanner l'environnement en 2D ou 3D. Ex. : RPLIDAR A1	Caméras (vision) Capturent des images pour la détection d'objets, la reconnaissance, le suivi... Ex. : Caméra OV5647	Capteurs infrarouges (IR) Détectent la présence d'objets ou suivent des lignes. Ex. : TCRT5000	Capteurs inertiels (IMU) Mesurent l'accélération et la vitesse angulaire (9 axes). Ex. : MPU-9250	Capteurs de contact Détectent un contact physique ou une fin de course. Ex. : Fin de course	Capteurs environnementaux Mesurent des paramètres comme température, humidité, lumière... Ex. : DHT22
---	---	---	---	--	--	--

COMMENT FONCTIONNE UN CAPTEUR ?



EXEMPLE : DÉTECTION D'OBSTACLE AVEC ULTRASON



Formule de calcul :

$$d = \frac{v \times t}{2}$$

où :

- d : distance à l'obstacle (m)
- v : vitesse du son (≈ 343 m/s)
- t : temps aller-retour (s)
- 2 : car l'onde fait l'aller et le retour

CRITÈRES DE CHOIX D'UN CAPTEUR

	Précision	Conseil : ✓ Choisissez le bon capteur pour la tâche, pas forcément le plus cher. ✓ Combinez plusieurs capteurs pour des données plus fiables. ✓ Testez toujours dans les conditions réelles d'utilisation.
	Portée Capacité à mesurer avec exactitude.	
	Distance maximale de détection.	
	Temps de réponse Temps pour l'information.	
	Robustesse Résistance aux conditions réelles.	
	Taille et poids Adapté à la conception du robot.	
	Consommation Impact sur l'autonomie.	
	Coût Selon le budget du projet.	

COMPARAISON RAPIDE

Type	Portée	Précision	Avantages	Limites	Coût
Ultrason	2 – 400 cm	★★★★☆	Peu coûteux, simple à utiliser	Moins fiable sur surfaces molles	\$
LIDAR	0.1 – 12 m	★★★★★	Précis, scan 2D/3D, bonne portée	Plus cher, consommation	\$\$\$\$
Caméra	Variable	★★★★☆	Vision riche, reconnaissance	Dépend de la lumière	\$\$\$
IR (suiveur)	1 – 10 cm	★★★★☆	Rapide, idéal pour suivre des lignes	Courte portée, sensible couteur	\$
IMU	N/A	★★★★☆	Mesure orientation et mouvement	Dérive dans le temps	\$\$
Contact	Contact direct	★★★★★	Très fiable, simple à intégrer	Nécessite contact physique	\$
Environnemental	Selon capteur	★★★★☆	Surveille l'état de l'environnement	Moins utile pour la navigation	\$

APPLICATIONS DES CAPTEURS EN ROBOTIQUE

Évitement d'obstacles Le robot détecte un obstacle et change de direction.	Suivi de ligne Les capteurs IR détectent la ligne et maintiennent le robot sur son trajet.	Cartographie et navigation Le LiDAR crée une carte de l'environnement pour naviguer autonomement.	Reconnaissance d'objets La caméra détecte et reconnaît des objets pour prendre des décisions.	Stabilisation et orientation L'IMU mesure les mouvements pour stabiliser ou équilibrer le robot.	Surveillance de l'environnement Les capteurs mesurent la température, l'humidité et la lumière.
--	--	---	---	--	---



BONNES PRATIQUES

- Calibrez vos capteurs avant utilisation.
- Filtrez les données pour éliminer le bruit.
- Combinez plusieurs capteurs pour plus de fiabilité.
- Protégez les capteurs de la poussière et des chocs.



À RETENIR

Les capteurs donnent au robot la perception nécessaire pour comprendre son environnement. Un bon choix et une bonne utilisation des capteurs sont essentiels pour créer des robots intelligents, efficaces et sécurisés.

2

CHAPITRE 2

Électronique
fondamentale



L'électronique est le langage qui permet au robot de penser et d'agir.

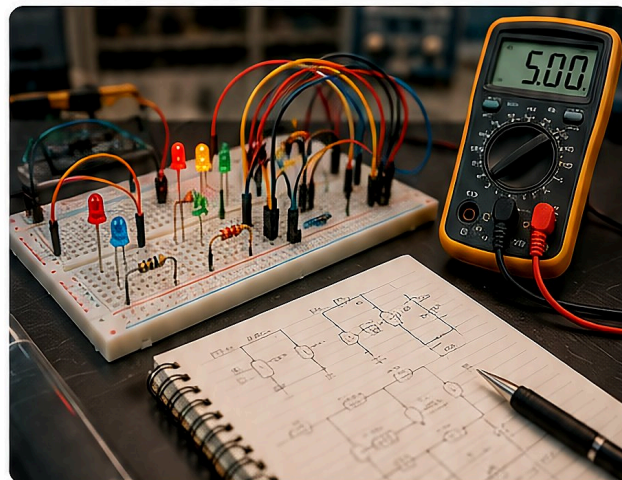
2.1

INTRODUCTION À L'ÉLECTRONIQUE

Le cœur électrique du robot

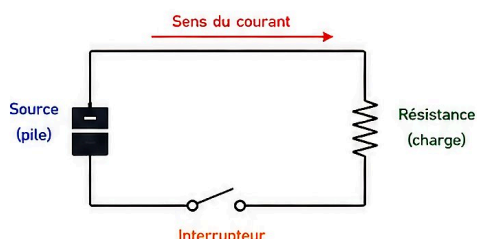
L'électronique permet de contrôler l'énergie électrique et les signaux dans un circuit pour réaliser des fonctions utiles.

Dans ce chapitre, nous allons découvrir les composants de base, leurs symboles, et comment ils sont utilisés dans les circuits des robots.



QU'EST-CE QU'UN CIRCUIT ÉLECTRIQUE ?

Un circuit électrique est un chemin fermé par lequel circule le courant électrique.
Il est constitué d'une source, de conducteurs et de composants.



GRANDEURS ÉLECTRIQUES DE BASE

- V** Tension (V) :
Différence de potentiel électrique entre deux points.
- I** Courant (A) :
Débit de charges électriques qui circulent dans un circuit.
- R** Résistance (Ω) :
Opposition au passage du courant.
Loi d'Ohm : $V = I \times R$

UNITÉS COURANTES

- Tension : volt (V)
- Courant : ampère (A)
- Résistance : ohm (Ω)
- Puissance : watt (W)



ASTUCE

Toujours vérifier les unités !
Elles évitent les erreurs dans vos montages.

COMPOSANTS ÉLECTRONIQUES DE BASE

Résistance



Limite le courant dans un circuit.

LED (diode électroluminescente)



Émet de la lumière lorsqu'elle est polarisée en direct.

Diode



Laisse passer le courant dans un seul sens.

Condensateur



Stocke des charges électriques.

Transistor (NPN)



Amplifie ou commute le courant.

Circuit intégré (CI)



Regroupe plusieurs composants dans un seul boîtier.

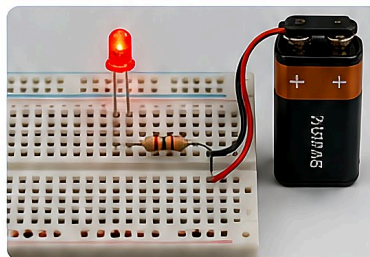
LECTURE D'UN SCHÉMA ÉLECTRIQUE

Les schémas utilisent des symboles normalisés pour représenter les composants et leurs connexions.

Composant	Symbole	Composant	Symbole
Pile		Condensateur	
Résistance		Interrupteur	
LED		Transistor NPN	
Diode		Masse (GND)	

EXEMPLE : FAIRE CLIGNOTER UNE LED

Un montage simple pour commencer !



Étapes :

- 1 Connecter la borne + de la pile à la résistance.
- 2 Connecter l'autre extrémité de la résistance à la LED (anode +).
- 3 Connecter la cathode (-) de la LED à la borne - de la pile.
- 4 La LED s'allume !



Rappel : Toujours utiliser une résistance avec une LED pour la protéger.

OUTILS INDISPENSABLES



Multimètre

Fer à souder

Étain

Pince coupante

Breadboard



À RETENIR

- ✓ L'électronique permet de contrôler l'énergie et les signaux.
- ✓ Un circuit est composé d'une source, de conducteurs et de composants.
- ✓ Les grandeurs de base sont : tension (V), courant (I) et résistance (R).
- ✓ Connaître les composants et leurs symboles est essentiel.
- ✓ La pratique et la sécurité sont les clés de la réussite !

2

CHAPITRE 2

Électronique
fondamentale



Comprendre
l'électricité est
essentiel pour
alimenter, contrôler
et faire fonctionner
les robots.

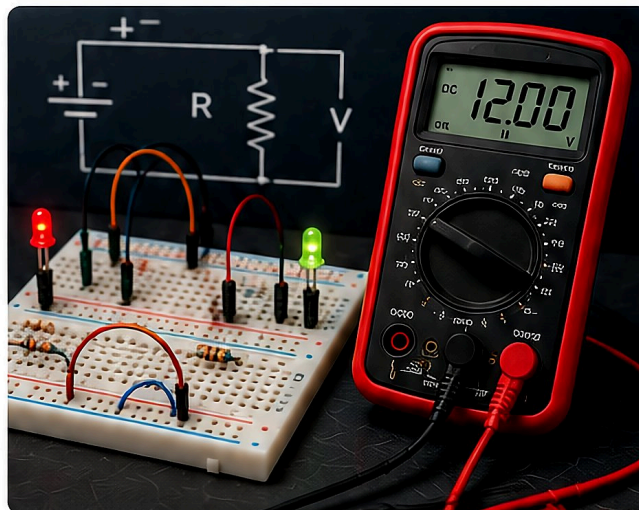
2.2

LES GRANDEURS ÉLECTRIQUES FONDAMENTALES

Comprendre les bases

Pour concevoir et analyser des circuits électroniques, il est indispensable de comprendre les grandeurs électriques fondamentales et leurs relations.

Ces grandeurs permettent de décrire le comportement de l'électricité dans un circuit et sont la base de toute électronique.



1. TENSION (V)

La tension est la différence de potentiel électrique entre deux points d'un circuit.

Unité : volt (V)

Symbole : V

Représentation :



2. COURANT (I)

Le courant électrique est le déplacement ordonné de charges électriques dans un conducteur.

Unité : ampère (A)

Symbole : I

Représentation :



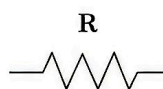
3. RÉSISTANCE (R)

La résistance s'oppose au passage du courant électrique.

Unité : ohm (Ω)

Symbole : R

Représentation :



4. PUISSANCE (P)

La puissance est l'énergie consommée ou fournie par unité de temps.

Unité : watt (W)

Symbole : P

Relation :

$$P = V \times I$$

LOI D'OHM

Dans un circuit résistif, la tension, le courant et la résistance sont liés par la loi d'Ohm :

$$V = I \times R$$

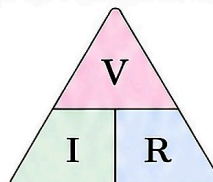
V : tension (V)

I : courant (A)

R : résistance (Ω)

TRIANGLE DE LA LOI D'OHM

Ce triangle aide à mémoriser les relations entre les grandeurs.



Formules dérivées :

- $V = I \times R$
- $I = \frac{V}{R}$
- $R = \frac{V}{I}$

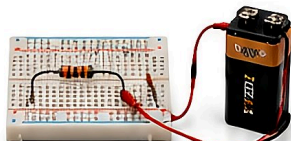
EXEMPLES NUMÉRIQUES

Exemple 1 : Calcul de courant

Une résistance de 100Ω est alimentée par une tension de 9 V.

Quel est le courant ?

$$I = \frac{V}{R} = \frac{9}{100} = 0,09 \text{ A}$$



Exemple 2 : Calcul de résistance

Un courant de 0,5 A traverse une résistance soumise à une tension de 12 V.

Quelle est la résistance ?

$$R = \frac{V}{I} = \frac{12}{0,5} = 24 \Omega$$



Exemple 3 : Calcul de puissance

Un moteur consomme un courant de 1,2 A sous une tension de 12 V.

Quelle est la puissance consommée ?

$$P = V \times I = 12 \times 1,2 = 14,4 \text{ W}$$



REMARQUES IMPORTANTES

- ⚠ Vérifiez toujours les unités avant de faire un calcul.
- 🎯 La tension se mesure toujours en parallèle entre deux points.
- ⚡ Le courant se mesure en série dans le circuit.
- ⚙ Utilisez des appareils adaptés : multimètre, alimentation régulée, résistances de puissance, etc.

OUTILS DE MESURE ET D'ESSAI

Multimètre



Mesure la tension,
le courant et la résistance.

Alimentation DC



Fournit une tension
continue réglable.

Ampèremètre



Mesure le courant
dans un circuit.

Voltmètre



Mesure la tension
entre deux points.

Ohmmètre



Mesure directement
la résistance.



À RETENIR

- La tension pousse le courant.
- Le courant transporte l'énergie.
- La résistance limite le courant.
- La puissance est liée à la consommation d'énergie.
- La loi d'Ohm est la clé pour analyser les circuits électriques.

2

CHAPITRE 2

Électronique
fondamentale



Communiquer
avec les
composants
électroniques
à l'aide des
symboles et des
schémas.

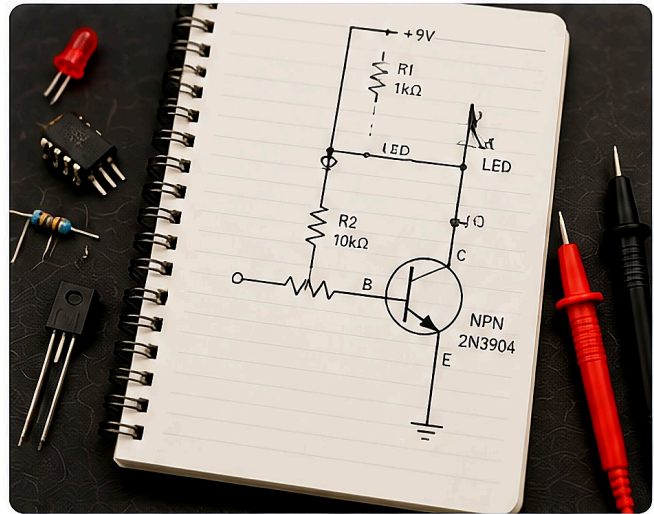
2.3

SYMBOLES ÉLECTRIQUES

Le langage universel des circuits

Les symboles électriques sont des représentations graphiques normalisées utilisées pour décrire les composants et les connexions dans un circuit.

Ils permettent de comprendre et de construire des schémas clairs, précis et compréhensibles partout dans le monde.

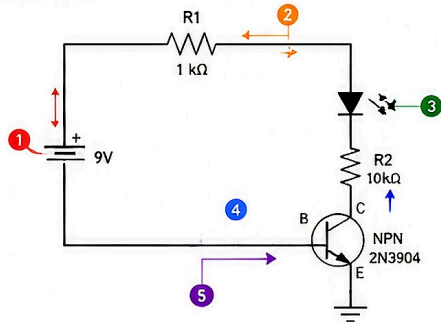


SYMBOLES DES COMPOSANTS LES PLUS COURANTS

Composant	Symbole	Composant	Symbole	Composant	Symbole
Pile / Batterie		Condensateur (polarisé)		Transistor NPN	
Source de tension continue		Diode		Transistor PNP	
Masse (Terre)		LED		Circuit intégré (CI)	
Résistance		Interrupteur (ouvert)		Moteur DC	
Résistance réglable (potentiomètre)		Interrupteur (fermé)		Relais	
Condensateur (non polarisé)		Fusible		Buzzer / Haut-parleur	

COMMENT LIRE UN SCHÉMA ÉLECTRIQUE ?

- Repérez la source d'alimentation.
- Suivez le chemin du courant conventionnel : du + vers le -.
- Identifiez chaque composant grâce à son symbole.
- Analysez les connexions et les nœuds du circuit.
- Comprenez le fonctionnement global du circuit.

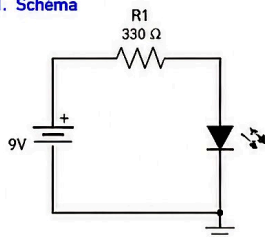


RÈGLES IMPORTANTES

- Toujours respecter la polarité des composants polarisés (diode, LED, condensateur, etc.).
- Les connexions qui se croisent sans point ne sont pas reliées.
- La masse (terre) est le point de référence du circuit.
- Éteignez toujours l'alimentation avant de modifier un circuit.
- Vérifiez vos branchements avant de mettre sous tension.

EXEMPLE : SCHÉMA D'UN CIRCUIT SIMPLE AVEC LED

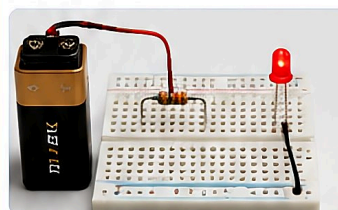
1. Schéma



2. Explication

- Le courant sort du +9V.
- Il traverse la résistance R1 qui limite le courant.
- Il alimente la LED, qui s'allume.
- Le courant retourne à la masse (-).

3. Montage réel



ASTUCE

Apprendre les symboles électriques rendra vos schémas plus professionnels et vos idées plus faciles à partager et à réaliser.



BONNES PRATIQUES

- Utilisez toujours les symboles normalisés dans vos schémas.
- Vérifiez deux fois vos connexions avant de mettre sous tension.
- Gardez vos schémas propres, lisibles et bien organisés.



À RETENIR

Les symboles électriques sont essentiels pour communiquer des idées de circuits de manière claire et universelle. Maîtrisez-les, et vous parlerez le langage des ingénieurs !

2

CHAPITRE 2

Électronique
fondamentale



Les lois fondamentales régissent le comportement des circuits électriques. Elles sont la base de toute analyse.

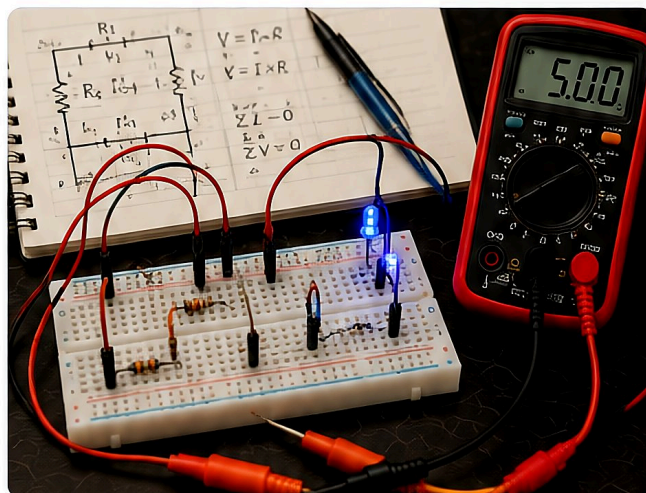
2.4

LOIS FONDAMENTALES DES CIRCUITS

Trois lois essentielles à connaître

Les circuits électriques obéissent à des règles précises. Trois lois fondamentales permettent d'analyser et de concevoir n'importe quel circuit.

Ces lois sont : la loi d'Ohm, les lois de Kirchhoff (courant et tension).

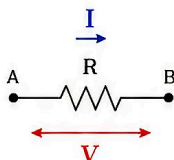


1 LOI D'OHM

Dans un circuit résistif, la tension (V) aux bornes d'un conducteur ohmique est égale au produit de la résistance (R) par le courant (I) qui le traverse.

$$V = I \times R$$

où : V = tension (V)
I = courant (A)
R = résistance (Ω)

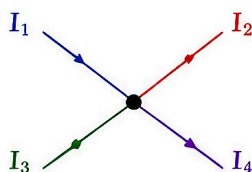


Plus la résistance est grande, plus le courant est faible pour une même tension.

2 LOI DES COURANTS DE KIRCHHOFF (LCK)

À un nœud (jonction) d'un circuit, la somme algébrique des courants est nulle.

$$\sum I_{\text{entrants}} = \sum I_{\text{sortants}} \quad \text{ou} \quad \sum I = 0$$

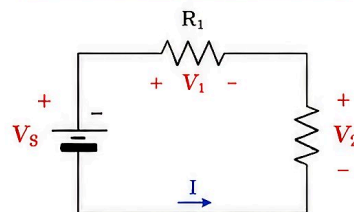


Le courant qui entre dans un nœud est égal au courant qui en sort.

3 LOI DES TENSIONS DE KIRCHHOFF (LTK)

Dans toute boucle fermée d'un circuit, la somme algébrique des tensions est nulle.

$$\sum V_{\text{élevations}} = \sum V_{\text{chutes}} \quad \text{ou} \quad \sum V = 0$$



La somme des hausses de tension est égale à la somme des chutes de tension.

EXEMPLES D'APPLICATION

Exemple 1 : Loi d'Ohm

Une résistance de 220 Ω est alimentée par une tension de 12 V. Quel courant la traverse ?

Solution :

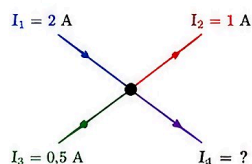
$$I = \frac{V}{R} = \frac{12}{220} = 0,0545 \text{ A}$$

soit 54,5 mA.



Exemple 2 : LCK

Au nœud ci-dessous, on a :
 $I_1 = 2 \text{ A}$, $I_2 = 1 \text{ A}$, $I_3 = 0,5 \text{ A}$ (sortant).
Calculer I_4 .



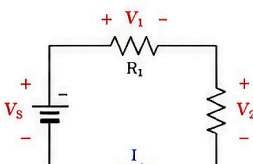
Solution :

$$I_1 + I_2 = I_3 + I_4$$

$$2 + 1 = 0,5 + I_4 \Rightarrow I_4 = 2,5 \text{ A}$$

Exemple 3 : LTK

Dans la boucle ci-dessous :
 $V_s = 12 \text{ V}$, $V_1 = 4 \text{ V}$
Trouver V_2 .



Solution :

$$V_s = V_1 + V_2 \Rightarrow V_2 = V_s - V_1$$

$$V_2 = 12 - 4 = 8 \text{ V}$$

REMARQUES IMPORTANTES



Respectez toujours les sens des courants et des polarités lors de l'analyse.



Choisissez un sens de référence et soyez cohérent dans tout le circuit.

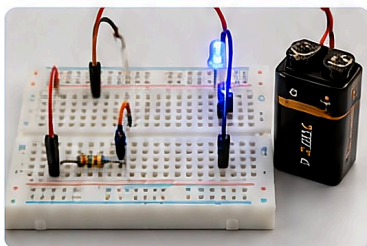


Les lois de Kirchhoff peuvent être appliquées à n'importe quel circuit, quelle que soit sa complexité.



Ces lois sont utilisées dans tous les domaines : électronique, robotique, automatisation, énergie, télécommunications, etc.

APPLICATION PRATIQUE : ANALYSE D'UN CIRCUIT SIMPLE



- 1 Identifier les éléments du circuit (source, résistances, LED, etc.).
- 2 Identifier les nœuds et les boucles.
- 3 Appliquer la LCK aux nœuds.
- 4 Appliquer la LTK aux boucles.
- 5 Utiliser la loi d'Ohm pour trouver les courants ou tensions inconnus.



À RETENIR

- ✓ Loi d'Ohm : $V = I \times R$
- ✓ Loi des courants de Kirchhoff : $\sum I = 0$ à un nœud
- ✓ Loi des tensions de Kirchhoff : $\sum V = 0$ dans une boucle
- ✓ Ces lois permettent d'analyser et de concevoir des circuits fiables.
- ✓ La pratique régulière est la clé de la maîtrise !

2

CHAPITRE 2

Électronique
fondamentale



La résistance est l'un des composants les plus utilisés dans les circuits électroniques.

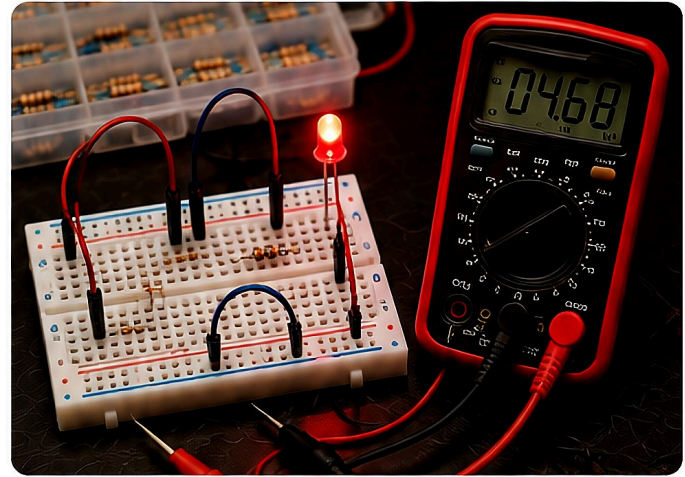
2.5

LES RÉSISTANCES

Contrôler et limiter le courant

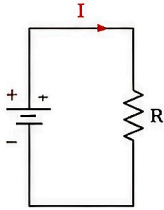
Une résistance limite le courant, divise la tension et protège les composants sensibles.

Elle est caractérisée par sa valeur en ohms (Ω) et sa puissance maximale.



1. QU'EST-CE QU'UNE RÉSISTANCE ?

Une résistance s'oppose au passage du courant électrique. Elle se mesure en ohms (Ω).



Loi d'Ohm :

$$V = I \times R$$

V : tension en volts (V)

I : courant en ampères (A)

R : résistance en ohms (Ω)

2. TYPES DE RÉSISTANCES

Résistances fixes



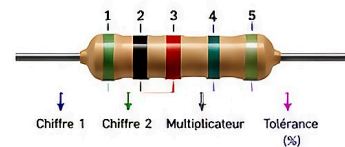
Valeur fixe, non modifiable.
Ex. : 220 Ω , 1 k Ω , 10 k Ω

Résistances variables



Valeur modifiable.
Ex. : potentiomètre, trimmer

3. CODE DES COULEURS DES RÉSISTANCES



Couleur	Chiffre 1	Chiffre 2	Multiplicateur	Tolérance (%)
Noir	0	0	$\times 10^0$	–
Marron	1	1	$\times 10^1$	± 1
Rouge	2	2	$\times 10^2$	± 2
Orange	3	3	$\times 10^3$	–
Jaune	4	4	$\times 10^4$	–
Vert	5	5	$\times 10^5$	± 0.5
Bleu	6	6	$\times 10^6$	± 0.25
Violet	7	7	$\times 10^7$	± 0.1
Gris	8	8	$\times 10^8$	± 0.05
Blanc	9	9	$\times 10^9$	–
Or	–	–	$\times 10^{-1}$	± 5
Argent	–	–	$\times 10^{-2}$	± 10

4. PUISSANCE DISSIPÉE

Une résistance dissipe de la puissance sous forme de chaleur. Il faut choisir une résistance dont la puissance est supérieure à celle du circuit.

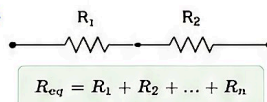
$$P = V \times I = I^2 \times R = \frac{V^2}{R}$$

Unités : watt (W)

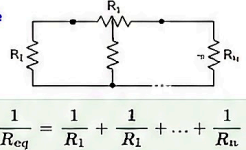
Les valeurs courantes : 1/8 W, 1/4 W, 1/2 W, 1 W, 2 W...

5. MONTAGES COURANTS

En série



En parallèle



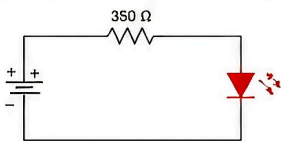
EXEMPLES D'APPLICATION

Exemple 1 : Limiter le courant d'une LED

LED rouge : $V_f = 2$ V Alimentation : $V_s = 9$ V
Courant souhaité : $I = 20$ mA

Calcul de la résistance :

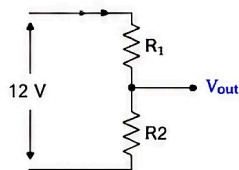
$$R = \frac{V_s - V_f}{I} = \frac{9 - 2}{0.02} = 350 \Omega$$



Utiliser une résistance standard de 360 Ω (1/4 W).

Exemple 2 : Diviseur de tension

$V_s = 12$ V, $R_1 = 2$ k Ω , $R_2 = 1$ k Ω



$$V_{out} = V_s \times \frac{R_2}{R_1 + R_2}$$

$$V_{out} = 12 \times \frac{1k}{2k + 1k} = 4$$
 V

Exemple 3 : Puissance dissipée

$R = 330 \Omega$ $I = 50$ mA

$$P = I^2 \times R$$

$$P = (0.05)^2 \times 330$$

$$P = 0.825$$
 W

Choisir une résistance d'au moins 1 W.



CONSEILS IMPORTANTS



Ne jamais dépasser la puissance maximale indiquée.



Vérifier la tolérance pour les circuits sensibles.



Une résistance chauffe : une chaleur excessive peut l'endommager.



Toujours couper l'alimentation avant de modifier un circuit.

OUTILS ET MATÉRIAUX



Multimètre



Kit de résistances



Breadboard



Fils de connexion



À RETENIR

- ✓ Une résistance limite le courant selon la loi d'Ohm : $V = I \times R$.
- ✓ Elle est identifiée par sa valeur (Ω) et sa puissance (W).
- ✓ Le code des couleurs permet de lire sa valeur facilement.
- ✓ Les résistances peuvent être en série ou en parallèle.
- ✓ Elles protègent les composants et contrôlent les signaux.

2

CHAPITRE 2

Électronique
fondamentale



Le condensateur stocke la charge électrique et la restitue lorsque nécessaire.

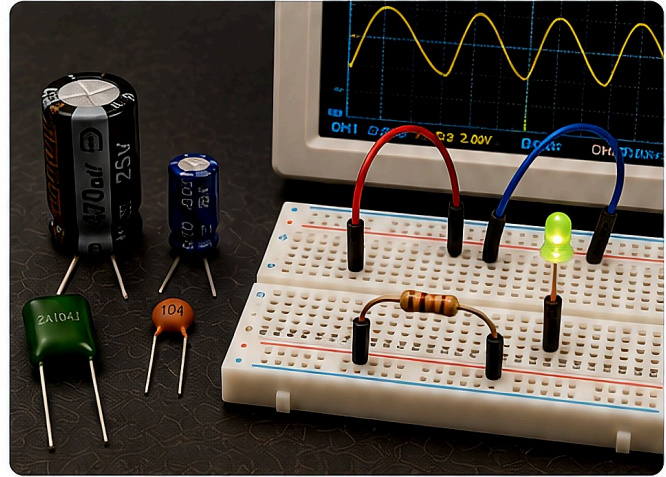
2.6

LES CONDENSATEURS

Stocker la charge et lisser la tension

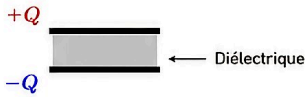
Un condensateur est un composant capable de stocker de l'énergie électrique sous forme de charge électrostatique dans un champ électrique.

Il est largement utilisé pour le filtrage, le découplage, la temporisation et le stockage d'énergie.



1. QU'EST-CE QU'UN CONDENSATEUR ?

Un condensateur est composé de deux plaques conductrices séparées par un matériau isolant (diélectrique). Il stocke des charges opposées sur ses plaques.



$$\text{Capacité : } C = \frac{Q}{V}$$

C : capacité en farads (F)
 Q : charge en coulombs (C)
 V : tension en volts (V)

2. SYMBOLE ÉLECTRIQUE

Le symbole d'un condensateur dans un schéma électrique est :

Non polarisé



Polarisé



Les condensateurs polarisés (électrolytiques) doivent être branchés dans le bon sens.

3. UNITÉS ET VALEURS

L'unité de capacité est le farad (F).
En pratique, on utilise :

- $1 \text{ nF} = 10^{-9} \text{ F}$ (nanofarad)
- $1 \text{ }\mu\text{F} = 10^{-6} \text{ F}$ (microfarad)
- $1 \text{ mF} = 10^{-3} \text{ F}$ (millifarad)

Valeur	Code
10 pF	100
100 pF	101
1 nF	102
10 nF	103
100 nF	104
1 μF	105
10 μF	106
100 μF	107

4. TYPES DE CONDENSATEURS



Céramique
Petite valeur, non polarisé, utilisé pour le filtrage haute fréquence.



Électrolytique
Grande capacité, polarisé, utilisé pour le filtrage et le stockage d'énergie.



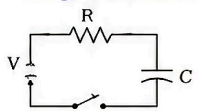
Film (polyester)
Bonne stabilité, faible perte, utilisé dans les circuits audio et de puissance.



Tantale
Compact et stable, polarisé.

5. COMPORTEMENT DANS UN CIRCUIT

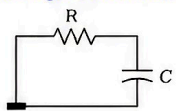
Charge du condensateur



$$V_C(t) = V \left(1 - e^{-\frac{t}{RC}} \right)$$

La tension aux bornes du condensateur augmente progressivement.

Décharge du condensateur



$$V_C(t) = V_0 e^{-\frac{t}{RC}}$$

La tension aux bornes du condensateur diminue progressivement.

6. ÉNERGIE STOCKÉE

L'énergie stockée dans un condensateur est :

$$E = \frac{1}{2} C V^2$$

où : E : énergie en joules (J)
 C : capacité en farads (F)
 V : tension en volts (V)

7. UTILISATIONS COURANTES

- Filtrage : lisser une tension continue.
- Découplage : éliminer le bruit dans les circuits électroniques.
- Temporisation : créer des délais de temps.
- Stockage d'énergie : alimentation de secours, flash, etc.

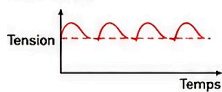


EXEMPLES D'APPLICATION

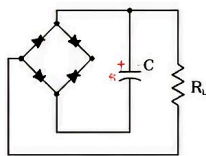
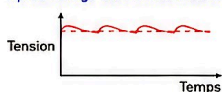
Exemple 1 : Filtrage d'une tension redressée

Le condensateur réduit l'ondulation d'une tension continue.

Avant filtrage

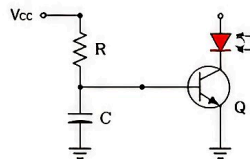


Après filtrage avec condensateur



Exemple 2 : Temporisation (circuit RC)

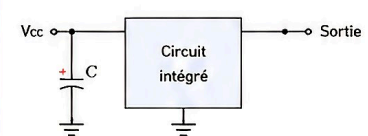
Le condensateur se charge et la LED s'allume après un délai.



Le temps d'allumage dépend de R et C :
 $t \approx 3 \times R \times C$

Exemple 3 : Découplage

Le condensateur élimine les parasites haute fréquence.



Placer le condensateur près du circuit intégré pour stabiliser l'alimentation.

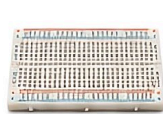
OUTILS ET MATÉRIAUX



Multimètre



Condensateurs



Breadboard



Fils de connexion



À RETENIR

- ✓ Le condensateur stocke des charges opposées sur ses plaques.
- ✓ La capacité se mesure en farads (F), μF , nF, etc.
- ✓ Respecter la polarité des condensateurs électrolytiques.
- ✓ Il est utilisé pour filtrer, découpler, temporiser et stocker de l'énergie.
- ✓ Un bon choix de capacité améliore la performance des circuits.

2

CHAPITRE 2

Électronique
fondamentale



La diode est un composant qui laisse passer le courant dans un seul sens et le bloque dans l'autre sens.

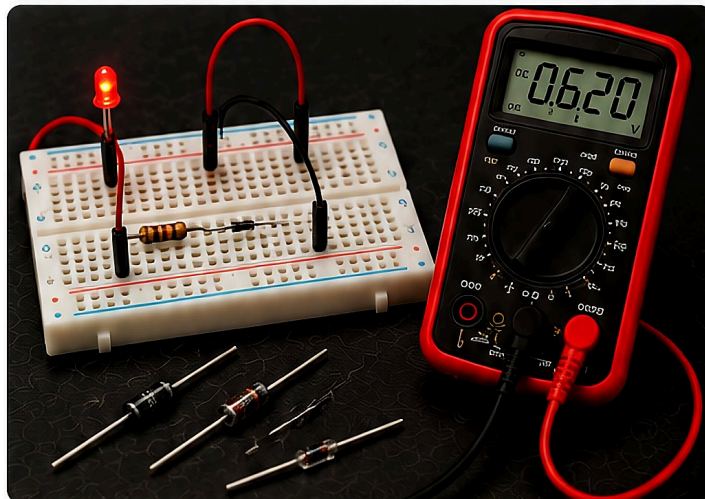
2.7

LES DIODES

Le courant dans un seul sens

Une diode est un composant électronique à deux bornes qui permet au courant de circuler principalement dans un seul sens.

Elle est utilisée pour le redressement, la protection des circuits, la détection de signaux et bien d'autres applications.



1 SYMBOLE ET STRUCTURE

Symbole électronique :

Anode (A) — Cathode (K)

Structure interne :

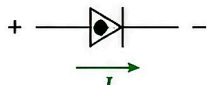
Anode (+) — P — N — Cathode (–)

La borne P est l'anode (+) et la borne N est la cathode (–).

2 FONCTIONNEMENT

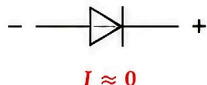
Polarisation directe (conduction)

La diode laisse passer le courant.



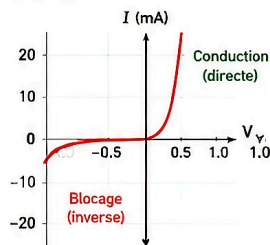
Polarisation inverse (blocage)

La diode bloque le courant.



3 CARACTÉRISTIQUE I-V

La courbe montre la relation entre le courant (I) et la tension (V) aux bornes de la diode.



Tension de seuil (V_s) $\approx 0,7$ V
(Typique pour les diodes en silicium)

4 TYPES DE DIODES

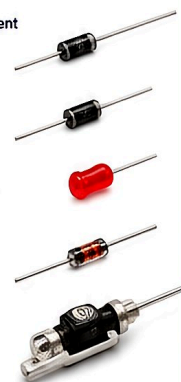
Diode de redressement (1N4007)

Diode Schottky (1N5819)

LED (diode électroluminescente)

Zener (diode de référence)

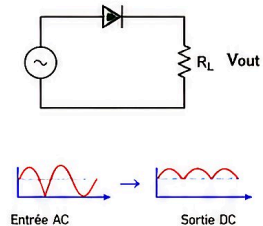
Diode rapide et de puissance



APPLICATIONS COURANTES

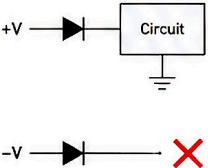
1. Redressement (AC → DC)

La diode transforme une tension alternative en tension continue (ondulée).



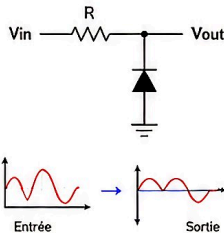
2. Protection contre l'inversion de polarité

La diode protège le circuit si l'alimentation est branchée à l'envers.



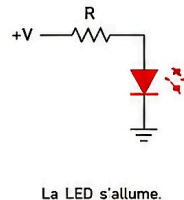
3. Écrêtage (clipping)

La diode limite la tension à une valeur donnée.



4. Indicateur lumineux

Une LED s'allume lorsqu'elle est polarisée directement.

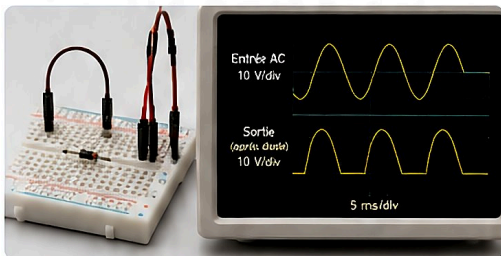


CONSEILS IMPORTANTS

- Toujours respecter la polarité d'une diode.
- Ne jamais appliquer une tension inverse supérieure à la tension maximale spécifiée.
- Utiliser la diode adaptée au courant et à la tension de votre circuit.
- Les LED nécessitent une résistance en série pour limiter le courant.

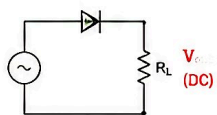
EXEMPLE PRATIQUE : REDRESSEMENT SIMPLE

Montage : Une diode 1N4007 redresse une tension alternative.



Résultat :

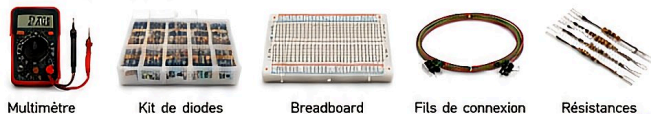
La sortie est une tension continue ondulée (demi-alternance positive seulement).



VALEURS TYPQUES DE DIODES COURANTES

Diode	Tension inverse max (V_R)	Courant direct max (I_F)	Tension de seuil (V_F) typique
1N4001	50 V	1 A	0,7 V
1N4007	1000 V	1 A	0,7 V
1N5819 (Schottky)	40 V	1 A	0,2 – 0,4 V
LED rouge	–	20 mA typ.	1,8 – 2,2 V
LED verte	–	20 mA typ.	2,0 – 3,2 V
Zener 5V1	5,1 V	500 mW	5,1 V

OUTILS ET MATÉRIAUX



À RETENIR

- La diode laisse passer le courant dans un seul sens.
- En polarisation directe : la diode conduit.
- En polarisation inverse : la diode bloque.
- Elle est utilisée pour le redressement, la protection, l'indication lumineuse et bien plus encore.

2

CHAPITRE 2

Électronique
fondamentale



Le transistor agit
comme un interrupteur
électronique ou comme
un amplificateur de
signal selon la façon
dont il est utilisé.

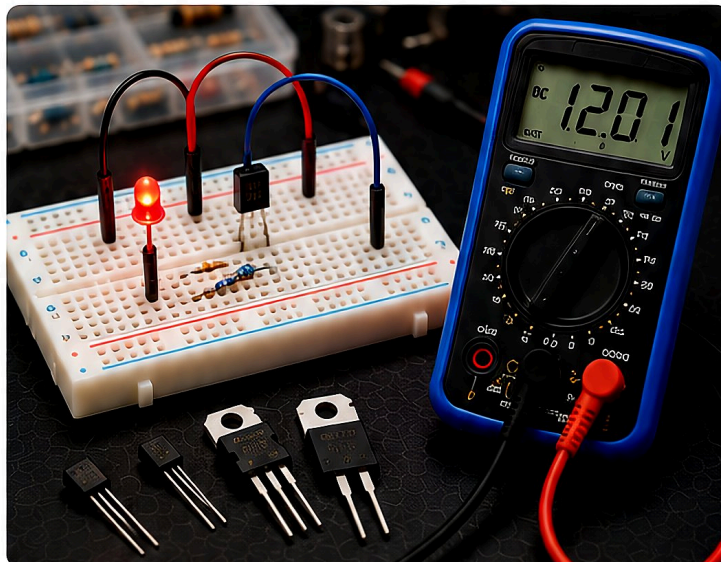
2.10

LES TRANSISTORS

Contrôler un grand courant avec un petit courant

Le transistor est un composant semi-conducteur
capable d'amplifier ou de commuter un courant
important à l'aide d'un courant beaucoup plus
faible appliqué sur sa base.

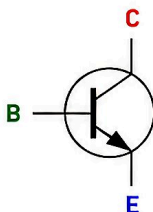
C'est l'un des composants les plus importants
de l'électronique moderne.



1 QU'EST-CE QU'UN TRANSISTOR ?

Un transistor est composé de trois
couches de semi-conducteur et de
trois bornes principales :

- Émetteur (E)
- Base (B)
- Collecteur (C)



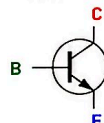
Dans ce chapitre, nous nous
concentrons sur le transistor NPN.

2 TYPES DE TRANSISTORS

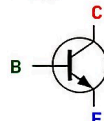
Selon leur structure :

BJT
(Bipolaire)

- NPN



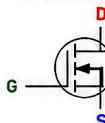
- PNP



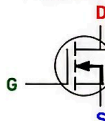
FET

(Effet de champ)

- N-Channel



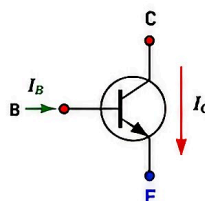
- P-Channel



Le BJT est idéal pour l'amplification
et la commutation. Le FET offre une
très grande impédance d'entrée.

3 FONCTIONNEMENT D'UN NPN

Le courant de base (I_B) contrôle le
courant entre collecteur et émetteur (I_C).



$$I_C = \beta \cdot I_B$$

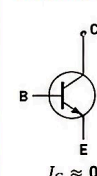
où β est le gain en courant
(généralement de 20 à 200
ou plus).

Un très petit courant de base
peut contrôler un courant beaucoup
plus grand entre C et E.

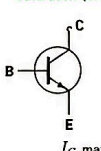
4 UTILISATIONS EN MODE

- Mode interrupteur

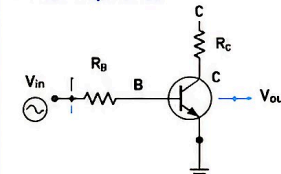
Coupeure (OFF)



Saturation (ON)

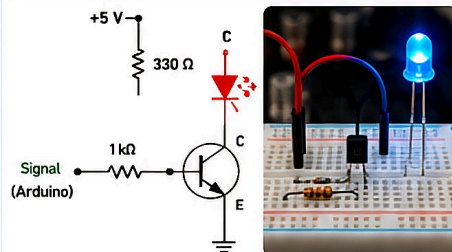


- Mode amplificateur



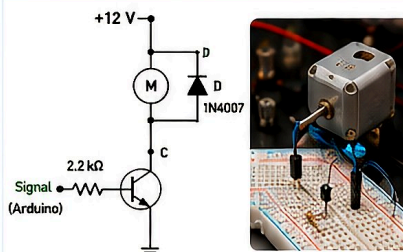
5 EXEMPLES PRATIQUES

a) Commande d'une LED



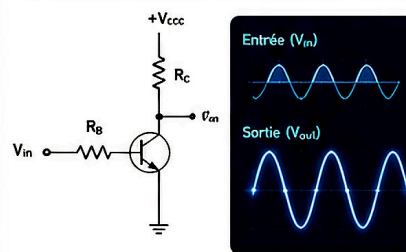
Le signal de la base allume ou éteint la LED.

b) Commande d'un moteur DC



Le transistor contrôle le moteur
et la diode protège le circuit.

c) Amplification de signal



Le transistor augmente l'amplitude
du signal de sortie.

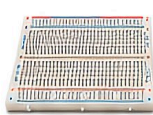
OUTILS ET MATÉRIAUX



Multimètre



Transistors
(NPN/PNP)



Breadboard



Fils de connexion



Résistances



À RETENIR

- ✓ Le transistor permet d'amplifier ou de commuter des courants.
- ✓ Il possède trois bornes : Émetteur (E), Base (B), Collecteur (C).
- ✓ Un petit courant de base contrôle un grand courant entre C et E.
- ✓ Utilisé partout : amplification, commutation, régulation, traitement de signal, commande de moteurs, etc.
- ✓ Comprendre son fonctionnement est essentiel en électronique et robotique.

2

CHAPITRE 2

Électronique fondamentale



Le relais permet de commander des charges importantes avec un faible courant de commande en assurant une isolation entre le circuit de commande et le circuit de puissance.

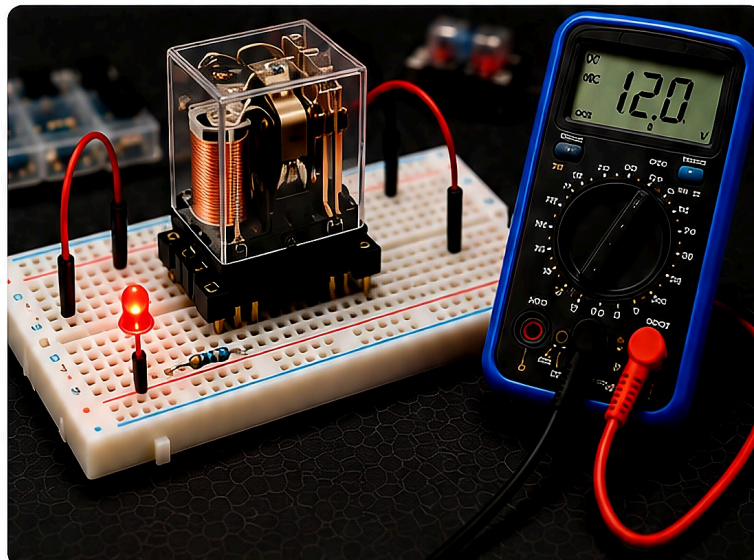
2.11

LES RELAIS

Commander de fortes charges avec un petit courant

Un relais est un composant électromécanique qui permet d'ouvrir ou de fermer un circuit électrique de puissance à l'aide d'un faible courant appliqué à sa bobine.

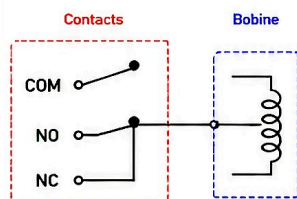
Il est largement utilisé pour la commande de moteurs, d'éclairages, de pompes, de systèmes d'automatisation et de protection.



1 QU'EST-CE QU'UN RELAIS ?

Un relais est constitué de deux parties principales :

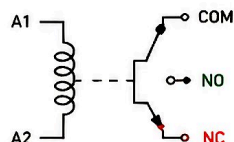
- La bobine (électroaimant)
- Les contacts (interrupteur mécanique)



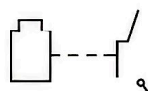
COM : Commun
NO : Normalement Ouvert
NC : Normalement Fermé

2 SYMBOLE ÉLECTRIQUE

Symbole complet :



Symbole simplifié :

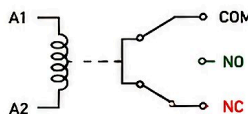


A1 – A2 : bornes de la bobine
COM : commun
NO : normalement ouvert
NC : normalement fermé

3 FONCTIONNEMENT

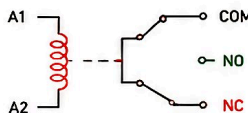
Relais au repos (bobine non alimentée)

Aucun courant dans la bobine.
Le contact NO est ouvert, le contact NC est fermé.



Relais activé (bobine alimentée)

Un courant circule dans la bobine.
Le contact NO se ferme, le contact NC s'ouvre.



4 CARACTÉRISTIQUES IMPORTANTES

- Tension de bobine : 5 V, 12 V, 24 V, 230 V...
- Courant de commande : faible (quelques mA)
- Courant de commutation : jusqu'à plusieurs dizaines d'ampères
- Isolation électrique entre commande et puissance
- Durée de vie mécanique : 10^5 à 10^7 cycles (selon le modèle)
- Temps d'activation : quelques ms
- Temps de relâchement : quelques ms



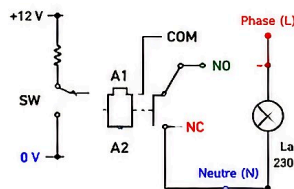
Différents types de relais

5 TYPES DE RELAIS

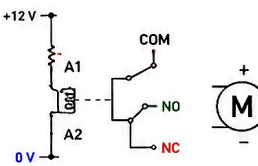
Relais électromécanique (EMR)	Relais statique (SSR)	Relais à verrouillage (latching)	Relais temporisé
Le plus courant, robuste et économique.	Sans contact mécanique, silencieux et rapide.	S'active par impulsion et reste dans l'état.	S'active ou se désactive après un délai réglable.

6 EXEMPLE DE MONTAGE

Commande d'une lampe 230 V AC avec un relais 12 V DC



Commande d'un moteur DC



Le relais isole le circuit de commande (basse tension) du circuit de puissance (haute tension ou courant élevé).

7 APPLICATIONS COURANTES

Commande de moteurs électriques	Contrôle d'éclairages et d'équipements	Pompes et ventilateurs	Domotique et automatisation	Protection et gestion des systèmes	Interfaces avec microcontrôleurs (Arduino, PLC...)

OUTILS ET MATÉRIELS

Multimètre	Relais 12 V	Module relais 1 canal	Breadboard	Fils de connexion	Alimentation 12 V DC



À RETENIR

- ✓ Le relais permet de commander de fortes charges avec un faible courant.
- ✓ Il possède une bobine (commande) et des contacts (puissance).
- ✓ Les contacts peuvent être NO, NC ou les deux.
- ✓ Il assure une isolation électrique efficace entre commande et puissance.
- ✓ Indispensable dans les systèmes de contrôle et d'automatisation.

2

CHAPITRE 2

Électronique
fondamentale



Les circuits intégrés regroupent plusieurs composants électroniques dans un seul boîtier pour réaliser des fonctions complexes de manière compacte et efficace.

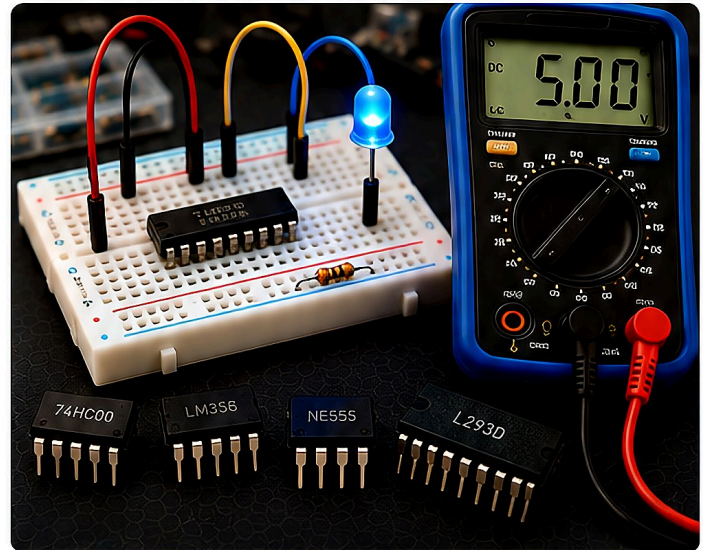
2.12

LES CIRCUITS INTÉGRÉS

De petites puces pour de grandes idées

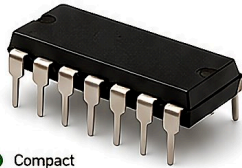
Un circuit intégré (CI ou IC) est un ensemble de composants électroniques miniaturisés (résistances, transistors, diodes, etc.) intégrés sur une puce de silicium.

Les circuits intégrés sont au cœur de presque tous les systèmes robotiques modernes : contrôle, traitement de données, communication, actionnement, capteurs, etc.



1 QU'EST-CE QU'UN CIRCUIT INTÉGRÉ ?

Un circuit intégré est une puce fabriquée sur du silicium. Il est placé dans un boîtier avec plusieurs broches pour se connecter au circuit.



- ✓ Compact
- ✓ Fiable
- ✓ Rapide
- ✓ Faible consommation
- ✓ Facile à utiliser

2 TYPES DE CIRCUITS INTÉGRÉS



Numériques

Portes logiques, compteurs, registres...



Analogiques

Amplificateurs, comparateurs, régulateurs...



Mixtes

Contiennent à la fois partie analogique et numérique.

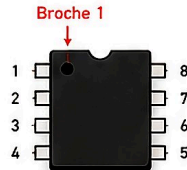


Microcontrôleurs

Contiennent un processeur, mémoire, entrées/sorties.

3 COMMENT LIRE UN CIRCUIT INTÉGRÉ

- Le petit point ou encoche indique la broche 1.
- On numérote ensuite dans le sens antihoraire (ou horaire selon le boîtier).



Vérifiez toujours la datasheet du CI avant de l'utiliser.

4 EXEMPLES DE CIRCUITS INTÉGRÉS COURANTS



NE555

Timer : génération d'impulsions, temporisations.



LM358

Double amplificateur opérationnel.



74HC00

Quadruple porte NAND (logique numérique).

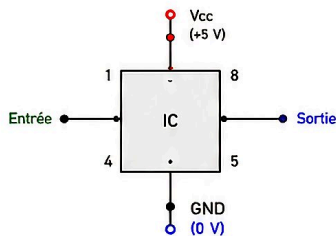


L293D

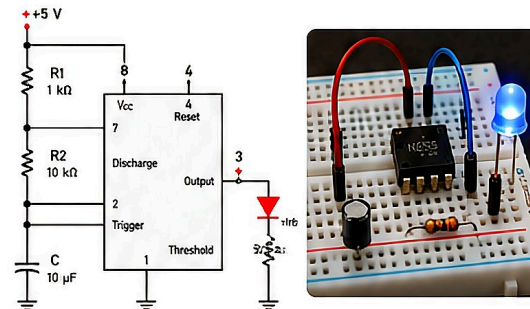
Pilote de moteurs DC (ponts en H).

5 COMMENT UTILISER UN CIRCUIT INTÉGRÉ

- Respecter l'alimentation (Vcc et GND).
- Respecter les broches d'entrée et de sortie.
- Utiliser des composants externes si nécessaire (résistances, condensateurs...).



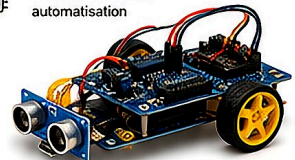
6 EXEMPLE PRATIQUE : NE555 EN MODE CLIGNOTANT LED



✓ Cette configuration fait clignoter la LED en continu.

7 APPLICATIONS EN ROBOTIQUE

- Pilotage de moteurs
- Traitement de signaux des capteurs
- Protection de circuits
- Communication et interfaces
- Contrôle logique et automatisation



EXEMPLES DE CIRCUITS INTÉGRÉS DANS UN SYSTÈME ROBOTIQUE

L298N / L293D

Pilotage des moteurs DC



LM358

Lecture des capteurs analogiques



Arduino (Microcontrôleur)

Traitement et contrôle intelligent



MPU6050

Accéléromètre et gyroscope (I2C)



OUTILS ET MATÉRIAUX



Multimètre


Kit de CI
(Diverses puces)


Breadboard



Fils de connexion


Datasheet
(documentation)


À RETENIR

- ✓ Les circuits intégrés permettent de réaliser des fonctions complexes avec peu de composants externes.
- ✓ Toujours vérifier le brochage et la tension d'alimentation.

- ✓ Ils sont indispensables dans tous les systèmes électroniques et robotiques.
- ✓ La lecture de la datasheet est essentielle !

2

CHAPITRE 2

Électronique
fondamentale

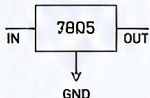
2.13

RÉGULATEURS DE TENSION

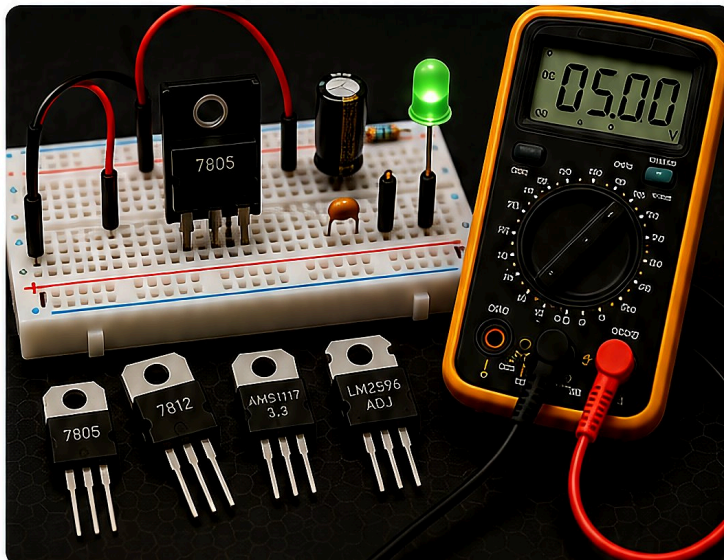
Alimenter vos circuits en toute sécurité

Un régulateur de tension est un circuit intégré qui délivre une tension de sortie constante, quelles que soient les variations de la tension d'entrée ou de la charge.

Il protège les composants sensibles des surtensions et des fluctuations, et assure un fonctionnement fiable des systèmes robotiques.



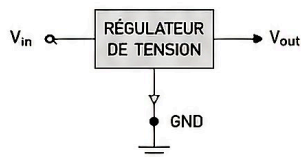
Les régulateurs de tension fournissent une tension stable en sortie, même si la tension d'entrée varie.



1 QU'EST-CE QU'UN RÉGULATEUR DE TENSION ?

Il maintient une tension de sortie (V_{out}) fixe, même si :

- La tension d'entrée (V_{in}) varie.
- Le courant de charge change.
- La température fluctue.



Sortie stable = circuits protégés et fonctionnement fiable.

2 TYPES DE RÉGULATEURS

Régulateurs linéaires

Simples, peu coûteux, faible bruit, mais moins efficaces.



Régulateurs à découpage (DC-DC)

Haute efficacité, idéals pour les systèmes à batterie.



3 RÉGULATEURS LINÉAIRES POPULAIRES

Référence	V_{out} fixe	Courant max
7805	5 V	1 A
7812	12 V	1 A
7815	15 V	1 A
7905	-5 V	1 A
AMS1117-3.3	3.3 V	1 A

Prévoir un dissipateur thermique si $(V_{in} - V_{out})$ est élevé et que le courant est important.

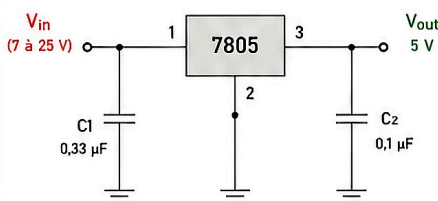
4 BROCHAGE (EXEMPLE : 7805)



- 1 IN : Entrée
- 2 GND : Masse
- 3 OUT : Sortie

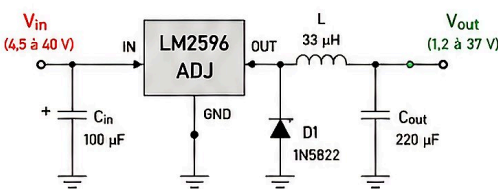
$V_{out} = 5$ V fixe
(avec le 7805)

5 CIRCUIT D'APPLICATION TYPIQUE (7805)



Les condensateurs C1 et C2 améliorent la stabilité du régulateur.

6 RÉGULATEUR À DÉCOUPAGE (EXEMPLE : LM2596)



Très efficace (> 85%), idéal pour les robots mobiles et les projets alimentés par batterie.

7 CONSEILS PRATIQUES

- Vérifiez toujours la tension d'entrée maximale du régulateur.
- Respectez le courant maximum.
- Utilisez un dissipateur thermique si nécessaire.
- Placez les condensateurs près du CI.
- Ne jamais inverser les connexions !



Une mauvaise connexion peut détruire le régulateur et les composants alimentés.

8 EXEMPLES D'UTILISATION EN ROBOTIQUE

Alimentation de capteurs



Fournit 5 V stables aux capteurs ultrasoniques, IR, IMU, etc.

Alimentation d'Arduino



Régule la tension pour protéger l'Arduino contre les variations.

Alimentation de servomoteurs (logique)



Fournit une tension stable pour la partie contrôle des servos.

Systèmes embarqués



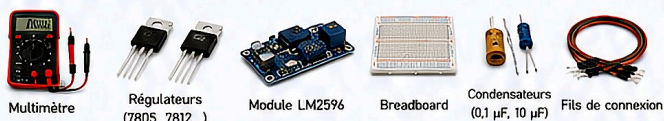
Alimente les microcontrôleurs et cartes embarquées.

Robots mobiles



Utilisé dans les modules de puissance et de gestion d'énergie.

OUTILS ET MATÉRIAUX



À RETENIR

- Le régulateur de tension fournit une tension stable en sortie.
- Il protège les circuits sensibles contre les variations.
- Les régulateurs linéaires sont simples, les à découpage sont efficaces.
- Toujours respecter le brochage et les valeurs limites.
- Indispensable dans tous les systèmes robotiques fiables.

2

CHAPITRE 2

Électronique
fondamentale



Sans une bonne alimentation, aucun circuit ne peut fonctionner correctement. Elle joue un rôle clé dans la fiabilité et la sécurité de votre robot.

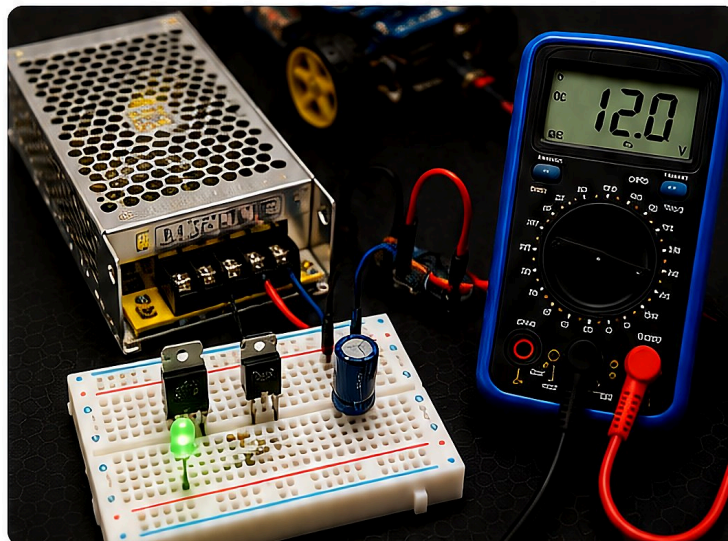
2.14

ALIMENTATION ÉLECTRIQUE

Fournir l'énergie à vos systèmes robotiques

Une alimentation électrique convertit l'énergie du secteur (AC) ou d'une batterie en tensions continues (DC) stables et adaptées aux composants électroniques.

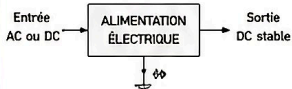
Elle est essentielle pour garantir la performance, la stabilité et la sécurité de tout système robotique.



1 QU'EST-CE QU'UNE ALIMENTATION ?

Une alimentation électrique fournit une tension continue (DC) stable à partir :

- Du secteur (AC 110 V / 230 V)
- D'une batterie
- D'un autre convertisseur

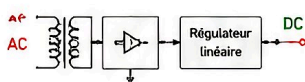


- ✓ Une tension stable = un robot fiable et des composants protégés.

2 TYPES D'ALIMENTATIONS

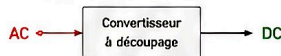
Alimentation linéaire

Simple, propre mais moins efficace (peu de rendement).



Alimentation à découpage (SMPS)

Très efficace, compacte et légère, utilisée dans la majorité des projets modernes.



SMPS : Switch Mode Power Supply

3 PARAMÈTRES IMPORTANTS

- Tension de sortie (V_{out})**
Ex : 5 V, 12 V, 24 V
- Courant maximal (I_{max})**
Doit être supérieur au besoin total.
- Puissance (W)**
 $P = V \times I$
- Rendement (%)**
Plus il est élevé, moins il y a de pertes.
- Ondulation (ripple)**
Doit être faible pour les circuits sensibles.

4 CONNEXIONS COURANTES

Alimentation DC simple



V_+ : positif (+)
 V_- : masse (- / GND)

Alimentation réglable



- ⚠ Toujours vérifier la polarité avant de connecter vos circuits !

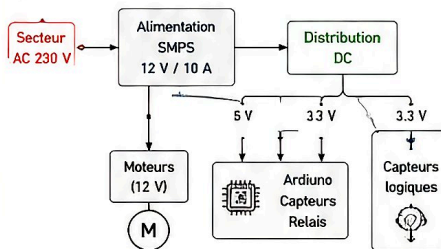
5 EXEMPLE : ALIMENTATION À DÉCOUPAGE 12 V / 5 A



Idéale pour alimenter moteurs, contrôleurs, LED, Arduino, etc.

Tension d'entrée	100 – 240 V AC
Tension de sortie	12 V DC
Courant max	5 A
Puissance max	60 W
Rendement	> 85 %
Protection	Court-circuit, Surcharge, Surtempérature
Refroidissement	Par convection

6 CIRCUIT D'ALIMENTATION TYPIQUE POUR ROBOT



- 💡 Séparer les alimentations des moteurs et de l'électronique sensibles réduit les interférences.

7 BONNES PRATIQUES

- ✓ Choisir une alimentation avec une marge de courant (20 à 30 % de plus).
- ✓ Utiliser des câbles adaptés au courant.
- ✓ Assurer une bonne ventilation.
- ✓ Protéger les sorties avec des fusibles si nécessaire.
- ✓ Vérifier régulièrement la température et la tension de sortie.

- ⚠ Une alimentation sous-dimensionnée peut provoquer des chutes de tension, des redémarrages ou endommager les composants.

8 APPLICATIONS EN ROBOTIQUE

Robots mobiles



Alimente moteurs, drivers et cartes de contrôle.

Bras robotiques



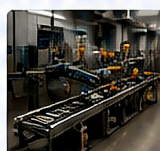
Fournit la puissance aux actionneurs.

Drones



Convertit la batterie LiPo en tensions stables.

Automatisation



Alimente capteurs, relais et automates.

Prototypage



Fournit une énergie stable pour tester vos circuits.

OUTILS ET MATÉRIAUX



Alimentation SMPS



Multimètre



Fusible



Bornier DC



Câbles



Connecteurs DC



À RETENIR

- ✓ Une alimentation stable est essentielle pour la performance et la sécurité.
- ✓ Choisissez toujours une alimentation adaptée en tension et en courant.
- ✓ Les alimentations à découpage (SMPS) sont plus efficaces et compactes.
- ✓ Respectez la polarité et protégez vos circuits.
- ✓ Une bonne alimentation = un robot fiable et durable.



ASTUCE

Pour les projets sensibles, ajoutez des condensateurs de filtrage (100 μ F à 1000 μ F) près des charges pour réduire l'ondulation et stabiliser la tension.



2

CHAPITRE 2

Électronique
fondamentale



La breadboard (plaque d'essai) permet de prototyper rapidement des circuits sans soudure. Elle est idéale pour apprendre, tester et modifier des montages en toute flexibilité.

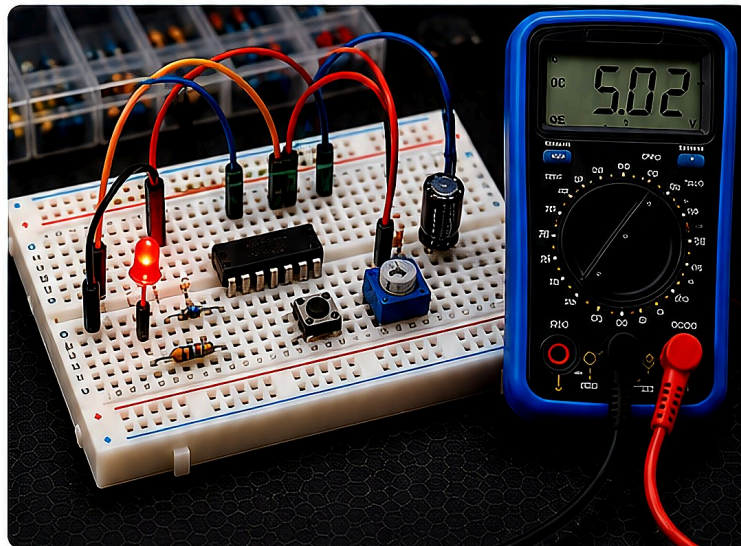
2.15

BREADBOARD ET PROTOTYPAGE

Construire, tester, améliorer

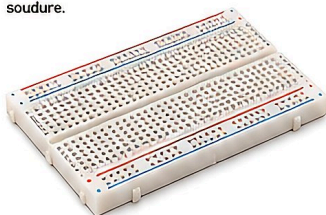
La breadboard est un outil indispensable pour concevoir et tester des circuits électroniques. Elle permet de connecter temporairement des composants et de modifier facilement les connexions.

C'est la première étape vers la création de systèmes robotiques fiables et fonctionnels.



1 QU'EST-CE QU'UNE BREADBOARD ?

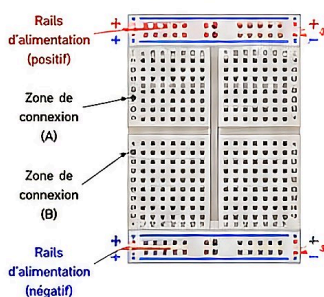
Une breadboard est une plaque en plastique percée de nombreux trous reliés entre eux par des connexions métalliques à l'intérieur. Elle permet d'assembler un circuit sans soudure.



- ✓ Réutilisable et économique
- ✓ Montage rapide et sans outil
- ✓ Idéale pour les tests et l'apprentissage
- ✓ Facile à modifier et à corriger

2 ANATOMIE DE LA BREADBOARD

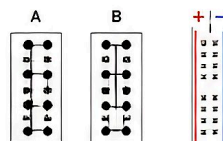
Vue de dessus



- Les trous d'une même colonne (A) sont connectés entre eux (sauf la fente centrale).
- Les rails latéraux sont utilisés pour l'alimentation.

3 COMMENT ÇA FONCTIONNE ?

Chaque colonne de 5 trous (A ou B) est électriquement connectée.



La fente centrale sépare les deux côtés pour accueillir les circuits intégrés (IC).

- Toujours prévoir une alimentation : V_{cc} (+) et GND (-).

4 RÈGLES À RESPECTER

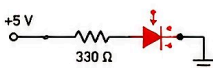
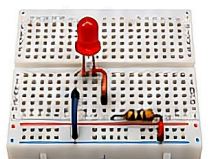
- Ne jamais court-circuiter V_{cc} et GND.
- Vérifier les polarités des composants (LED, condensateurs, etc.).
- Utiliser des fils de différentes couleurs pour plus de clarté.
- Tester chaque partie du circuit avant de l'intégrer au projet final.



Une mauvaise connexion peut endommager vos composants ou votre carte électronique !

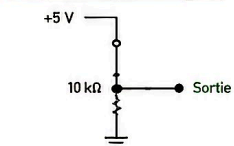
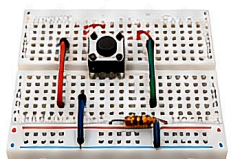
5 EXEMPLES DE MONTAGES SUR BREADBOARD

a) Allumer une LED



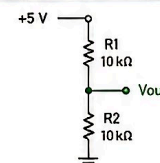
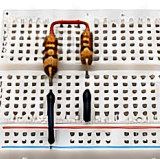
La LED s'allume lorsque le courant circule de V_{cc} vers GND.

b) Bouton poussoir (Push Button)



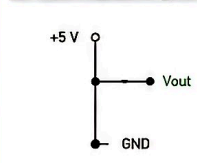
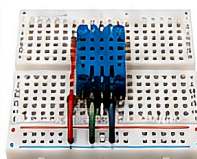
Le bouton envoie un signal haut (5 V) lorsqu'il est pressé.

c) Diviseur de tension



Permet d'obtenir une tension intermédiaire.

d) Test d'un capteur



Lecture de la sortie analogique d'un capteur.

6 ÉTAPES DE PROTOTYPAGE



Définir l'objectif du circuit
Que doit faire le circuit ?



Choisir les composants
Lister les composants nécessaires.



Réaliser le montage sur breadboard
Connecter selon le schéma.



Tester et vérifier
Mesurer les tensions et signaux.

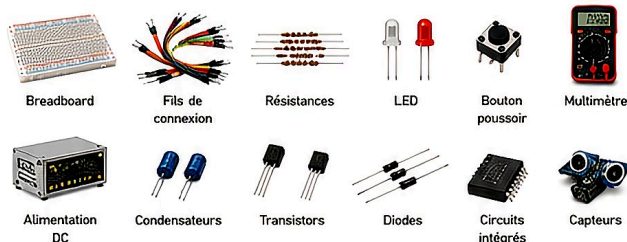


Corriger et améliorer
Apporter les modifications nécessaires.



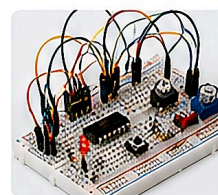
Passer à la version finale
Solder sur PCB ou utiliser une carte (Arduino, ESP32, etc.).

7 OUTILS ET MATÉRIAUX ESSENTIELS



8 BONS CONSEILS

- ✓ Commencez par des circuits simples et progressez étape par étape.
- ✓ Gardez votre montage propre et organisé.
- ✓ Étiquetez les fils et les parties importantes.
- ✓ Prenez des photos de vos montages pour référence future.
- ✓ La pratique régulière est la clé pour devenir un excellent roboticien !



Prototyper, c'est apprendre en testant !
C'est ainsi que naissent les meilleures innovations.

2

CHAPITRE 2 Électronique fondamentale



Une bonne alimentation électrique assure la stabilité, la performance et la sécurité de tous les circuits électroniques et systèmes robotiques.

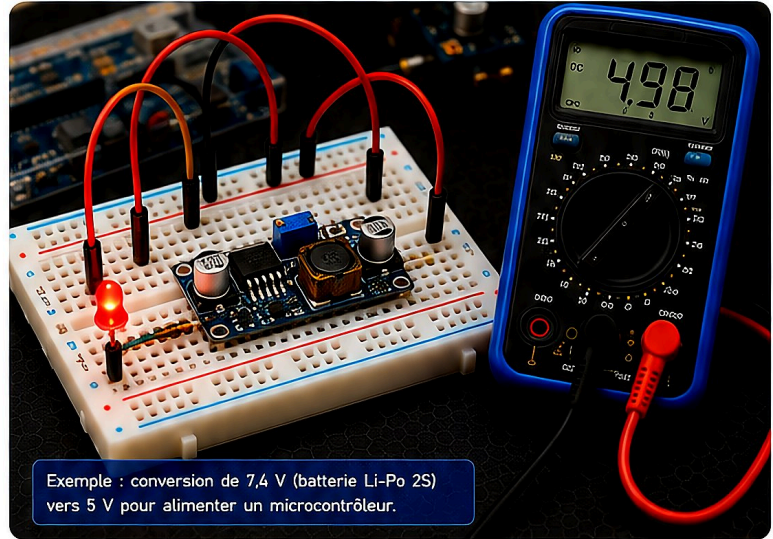
2.16

CONVERTISSEURS DC-DC

Adapter la tension pour chaque besoin

Les convertisseurs DC-DC permettent de convertir une tension continue (DC) en une autre tension continue différente. Ils sont essentiels dans les systèmes robotiques pour alimenter efficacement chaque module selon ses besoins.

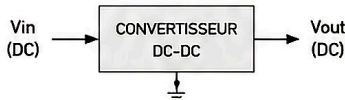
Ils offrent une meilleure efficacité énergétique et permettent de réduire la dissipation de chaleur comparé aux régulateurs linéaires.



Exemple : conversion de 7,4 V (batterie Li-Po 2S) vers 5 V pour alimenter un microcontrôleur.

1 QU'EST-CE QU'UN CONVERTISSEUR DC-DC ?

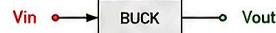
Un convertisseur DC-DC prend une tension continue d'entrée (V_{in}) et la convertit en une tension continue de sortie (V_{out}) plus faible, plus élevée ou inversée.



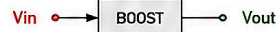
- ✓ Haute efficacité (80 à 95 % typique)
- ✓ Faible génération de chaleur
- ✓ Taille compacte et légère
- ✓ Idéal pour les systèmes à batterie

2 TYPES DE CONVERTISSEURS DC-DC

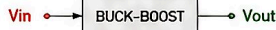
Abaisseur (Buck) – $V_{out} < V_{in}$



Élévateur (Boost) – $V_{out} > V_{in}$



Abaisseur-Élévateur (Buck-Boost) – $V_{out} < \text{ou} > V_{in}$



Inverser (Inverting) – polarité inversée



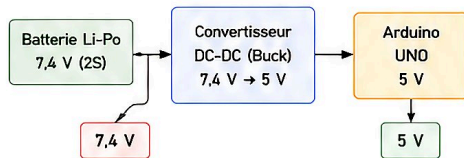
3 PARAMÈTRES IMPORTANTS

- Tension d'entrée (V_{in})**
Plage de tension acceptée par le module.
- Tension de sortie (V_{out})**
Tension fournie en sortie.
- Courant de sortie (I_{out})**
Courant maximal disponible.
- Rendement (η)**
Pourcentage d'énergie convertie efficacement.
- Protection intégrée**
Contre surintensité, court-circuit, surchauffe, etc.

4 EXEMPLES DE MODULES

- LM2596 (Buck)**
Entrée : 4 – 40 V
Sortie : 1,25 – 37 V
Courant max : 3 A
- MT3608 (Boost)**
Entrée : 2 – 24 V
Sortie : 5 – 28 V
Courant max : 2 A
- XL6009 (Buck-Boost)**
Entrée : 5 – 32 V
Sortie : 1,25 – 35 V
Courant max : 4 A
- LM2596 Inverter**
Entrée : 4 – 40 V
Sortie : -1,25 à -37 V
Courant max : 2 A

5 EXEMPLE PRATIQUE : ALIMENTER UN ARDUINO



Pourquoi utiliser un convertisseur ?

- L'Arduino nécessite 5 V stables.
- Le convertisseur fournit une tension stable même si la batterie varie.
- Protège l'Arduino et améliore l'autonomie.

6 CALCUL DE LA PUISSANCE

Pour choisir un convertisseur adapté :

$$P_{out} (W) = V_{out} (V) \times I_{out} (A)$$

Exemple :

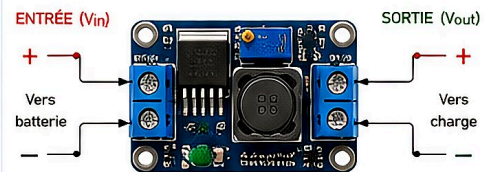
$$V_{out} = 5 \text{ V}, I_{out} = 2 \text{ A}$$

$$P_{out} = 5 \times 2 = 10 \text{ W}$$

Choisissez un convertisseur dont la puissance maximale (P_{max}) est supérieure à P_{out} avec une marge de sécurité (20 à 30 %).

Astuce : ne pas utiliser un module à pleine charge en continu.

7 CÂBLAGE DE BASE



⚠ Respectez toujours la polarité d'entrée et de sortie ! Une inversion peut endommager le module.

8 APPLICATIONS EN ROBOTIQUE

Alimentation des microcontrôleurs



Fournir une tension stable de 5 V ou 3,3 V.

Alimentation des servomoteurs



Convertir la tension de batterie pour chaque servo.

Alimentation des capteurs



Adapter la tension pour chaque type de capteur.

Systèmes embarqués



Fournir des tensions spécifiques à différents modules.

Robots mobiles



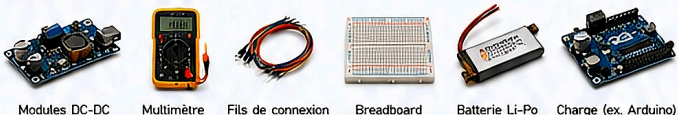
Gérer efficacement l'énergie pour prolonger l'autonomie.

Drones et UAV



Répartir et réguler l'énergie pour les contrôleurs de vol et les caméras.

9 OUTILS ET MATÉRIAUX



À RETENIR

- ✓ Les convertisseurs DC-DC adaptent la tension selon les besoins.
- ✓ Ils sont plus efficaces et compacts que les régulateurs linéaires.
- ✓ Choisissez un module avec une marge de courant et de puissance.
- ✓ Vérifiez toujours les polarités et les spécifications avant le branchement.
- ✓ Essentiels pour des systèmes robotiques fiables et performants.

2

CHAPITRE 2

Électronique
fondamentale

2.17

REDRESSEURS ET FILTRAGE

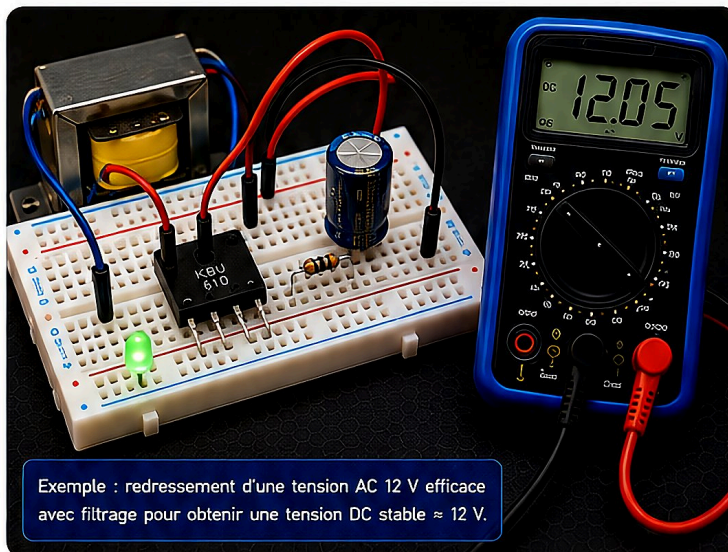
Transformer l'AC en DC propre et stable



Le redressement convertit une tension alternative (AC) en tension continue (DC) pulsée, qui devient plus stable après filtrage.

Les redresseurs permettent de convertir une tension alternative (AC) en tension continue (DC) pulsée. Le filtrage, réalisé avec des condensateurs, réduit l'ondulation pour obtenir une tension plus lisse et exploitable par les circuits électroniques.

Ces circuits sont essentiels dans les alimentations de robots, chargeurs, et de nombreux systèmes électroniques.



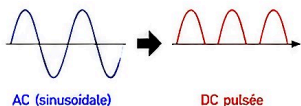
Exemple : redressement d'une tension AC 12 V efficace avec filtrage pour obtenir une tension DC stable ≈ 12 V.

1 POURQUOI REDRESSER ?

La plupart des appareils électroniques et des cartes de contrôle fonctionnent avec du courant continu (DC). Or, le secteur fournit une tension alternative (AC).

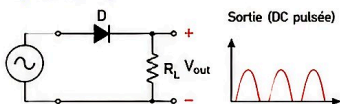
Le redressement permet donc :

- ✓ De convertir AC en DC.
- ✓ D'alimenter les circuits électroniques.
- ✓ De recharger les batteries.
- ✓ De réduire le bruit avec un filtrage.

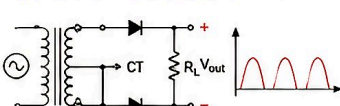


2 TYPES DE REDRESSEURS

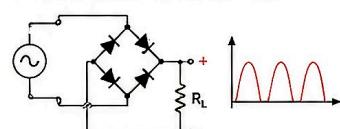
a) Demi-onde



b) Double alternance avec point milieu



c) Pont de diodes (redresseur en pont)



3 PONT DE DIODES

Le pont de diodes est le redresseur double alternance le plus utilisé. Il utilise 4 diodes pour redresser les deux alternances du signal AC.

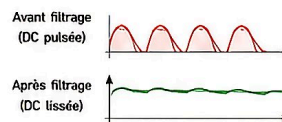
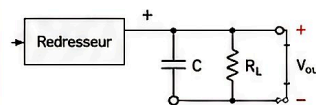


Caractéristiques courantes

Courant max	1 A à 10 A
Tension inverse max	100 V à 1000 V
Faible chute de tension	~ 1 V (2 diodes conduisent)

4 FILTRAGE AVEC CONDENSATEUR

Le condensateur se charge à la valeur crête de la tension redressée et se décharge lentement entre les crêtes, réduisant l'ondulation.



Plus la capacité (C) est grande, plus la tension est lisse.

5 CALCUL DE L'ONDULATION

L'ondulation (ripple) est approximée par :

$$V_{\text{ripple}} \approx \frac{I_{\text{load}}}{f \times C}$$

Où :

- I_{load} : courant de charge (A)
- f : fréquence du redressement (Hz)
(100 Hz pour double alternance à 50 Hz)
- C : capacité du condensateur (F)

Augmenter C ou diminuer I_{load} réduit l'ondulation.

6 EXEMPLE PRATIQUE

Transformer 12 V AC (efficace) en ≈ 12 V DC stable.



Résultat :

- ✓ Tension DC à vide $\approx 16,8$ V (crête) – 1,4 V (pont)
- ✓ Tension DC sous charge ≈ 12 V
- ✓ Ondulation réduite grâce au condensateur

7 CHOIX DU CONDENSATEUR

Capacité recommandée :

$$C (\mu\text{F}) \approx \frac{1000 \times I_{\text{load}} (\text{A})}{V_{\text{ripple}} (\text{V})}$$

Tension de travail :

Toujours choisir un condensateur dont la tension nominale est supérieure à la tension maximale attendue.

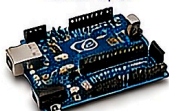
Exemple : pour 12 V DC, utiliser au moins 25 V.

8 PRÉCAUTIONS

- ⚠ Respecter la polarité du condensateur (+ vers le +).
- ⚠ Ne jamais court-circuiter la sortie du redresseur.
- ⚠ Utiliser des diodes avec une tension inverse et un courant max adaptés.
- ⚠ Décharger le condensateur avant toute manipulation.

9 APPLICATIONS EN ROBOTIQUE

Alimentation de cartes électroniques



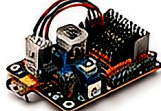
Fournit une tension DC stable pour les microcontrôleurs.

Chargeurs de batteries



Convertit l'AC du secteur en DC pour recharger les batteries.

Alimentation de moteurs



Alimente les drivers de moteurs DC ou servomoteurs.

Systèmes embarqués



Fournit une alimentation stable pour les cartes embarquées.

Bras robotiques



Alimente les actionneurs et capteurs de manière fiable.

Imprimantes 3D



Convertit l'AC en DC pour les cartes et moteurs pas à pas.

Drones et UAV



Alimente les contrôleurs de vol et capteurs de navigation.

10 OUTILS ET MATÉRIAUX



Transformateur AC



Pont de diodes (KBU610)



Condensateur électrolytique



Résistance (de charge)



Multimètre



Breadboard et fils

À RETENIR

- ✓ Le redresseur convertit l'AC en DC pulsée.
- ✓ Le condensateur de filtrage réduit l'ondulation pour une tension plus stable.
- ✓ Le pont de diodes est la solution la plus courante pour le redressement double alternance.
- ✓ Un bon filtrage assure la stabilité et la fiabilité des systèmes robotiques.

2

CHAPITRE 2

Électronique
fondamentale



Les transformateurs modifient les niveaux de tension en courant alternatif (AC) sans changer la fréquence. Ils sont essentiels pour adapter l'énergie aux besoins de nos circuits.

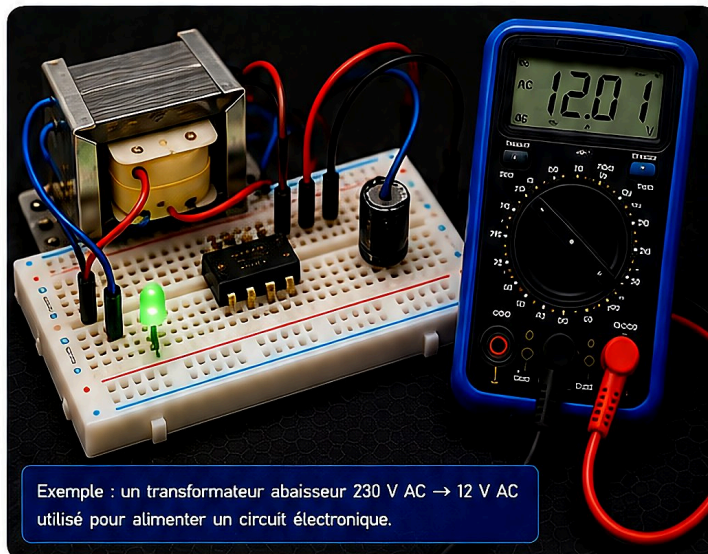
2.18

TRANSFORMATEURS

Adapter les tensions en courant alternatif

Un transformateur est un composant électrique passif qui permet d'augmenter (élever) ou de diminuer (abaisser) une tension alternative (AC) tout en conservant la même fréquence. Il fonctionne grâce à l'induction électromagnétique entre deux enroulements.

Les transformateurs sont largement utilisés dans les alimentations, les chargeurs, les équipements industriels et de nombreux systèmes robotiques.



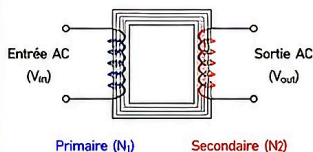
Exemple : un transformateur abaisseur 230 V AC → 12 V AC utilisé pour alimenter un circuit électronique.

1 QU'EST-CE QU'UN TRANSFORMATEUR ?

Le transformateur transfère de l'énergie électrique d'un circuit à un autre par induction magnétique.

Il ne fonctionne qu'en courant alternatif (AC) et permet de :

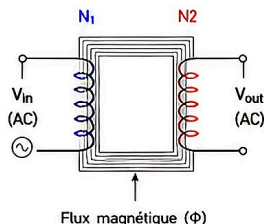
- ✓ Élever ou abaisser la tension
- ✓ Isoler électriquement les circuits
- ✓ Adapter une source aux besoins de l'application



Primaire (N_p) Secondaire (N_s)

2 PRINCIPE DE FONCTIONNEMENT

Un courant alternatif dans le primaire (N₁) crée un flux magnétique variable dans le noyau. Ce flux induit une tension dans le secondaire (N₂).



⚡ Pas de contact électrique direct entre primaire et secondaire → isolation électrique.

3 RAPPORT DE TRANSFORMATION

Le rapport entre les tensions est égal au rapport entre le nombre de spires :

$$\frac{V_s}{V_p} = \frac{N_s}{N_p}$$

Où :

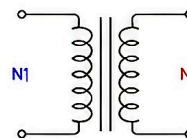
V_p : tension primaire (V)
V_s : tension secondaire (V)
N_p : nombre de spires du primaire
N_s : nombre de spires du secondaire

Types de transformateurs

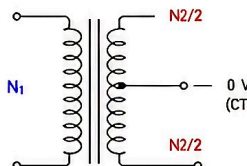
Type	Condition	Tension secondaire
Abaisseur	N _s < N _p	V _s < V _p
Élévateur	N _s > N _p	V _s > V _p
Isolateur	N _s = N _p	V _s = V _p

4 SYMBOLE ÉLECTRIQUE

Transformateur idéal



Transformateur avec prise médiane



5 EXEMPLE DE CALCUL

Un transformateur a N_p = 1000 spires et N_s = 200 spires. La tension primaire est de 230 V AC. Quelle est la tension secondaire ?

Solution :

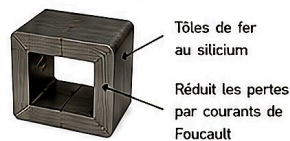
$$\frac{V_s}{V_p} = \frac{N_s}{N_p} \Rightarrow V_s = V_p \times \frac{N_s}{N_p}$$

$$V_s = 230 \times \frac{200}{1000} = 46 \text{ V AC}$$

Résultat : V_s = 46 V AC

6 TYPES DE NOYAUX

a) Noyau en tôles empilées (le plus courant)



Tôles de fer au silicium
Réduit les pertes par courants de Foucault

b) Noyau torique



Faible fuite magnétique
Compact et efficace

7 APPLICATIONS EN ROBOTIQUE

- Alimentation de cartes électroniques (Arduino, ESP32, microcontrôleurs)
- Chargeurs de batteries
- Isolation entre circuits de puissance et circuits de commande
- Entrée des alimentations à découpage
- Adaptation de capteurs ou de moteurs AC



8 BONNES PRATIQUES

- ✓ Utiliser un transformateur adapté à la puissance nécessaire (VA).
- ✓ Ne jamais dépasser la tension ou le courant nominal.
- ✓ Assurer une bonne ventilation si la puissance est élevée.
- ✓ Vérifier l'isolation si le transformateur alimente des circuits sensibles.
- ✓ Toujours mesurer la tension à vide avant de connecter une charge.



Un transformateur mal dimensionné peut chauffer, saturer ou être détruit.

9 EXEMPLES DE TRANSFORMATEURS



Transformateur abaisseur
230 V AC → 12 V AC
(1 A)



Transformateur élévateur
12 V AC → 230 V AC
(150 W)



Transformateur torique
2 × 15 V AC
(2 A)



Transformateur d'isolement
230 V AC → 230 V AC
(500 VA)



Mini transformateur PCB
230 V AC → 9 V AC
(5 VA)

10 À RETENIR



- ✓ Le transformateur fonctionne uniquement avec du courant alternatif (AC).
- ✓ Il permet d'élever, d'abaisser ou d'isoler les tensions.
- ✓ Le rapport des tensions dépend du nombre de spires.
- ✓ Indispensable pour alimenter et protéger les circuits en robotique.

2

CHAPITRE 2

Électronique
fondamentale



Le condensateur stocke l'énergie entre deux armatures séparées par un isolant (diélectrique). Plus sa capacité est grande, plus il peut stocker de charge.

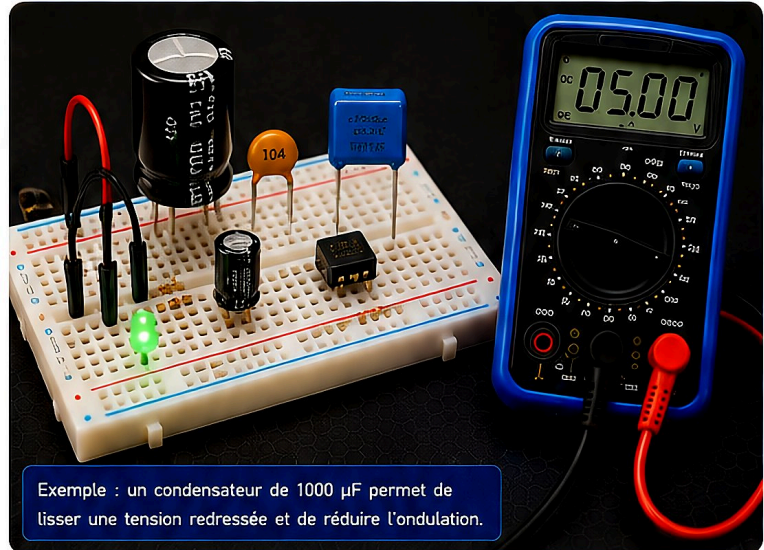
2.19

CONDENSATEURS ET STOCKAGE D'ÉNERGIE

Stocker, lisser et stabiliser la tension

Le condensateur est un composant électronique capable de stocker de l'énergie dans un champ électrique. Il est largement utilisé pour filtrer le bruit, lisser les tensions, temporiser des circuits et stabiliser les alimentations.

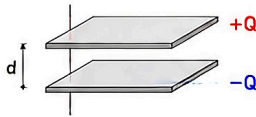
Comprendre son fonctionnement est essentiel pour construire des systèmes robotiques fiables et performants.



Exemple : un condensateur de 1000 μF permet de lisser une tension redressée et de réduire l'ondulation.

1 QU'EST-CE QU'UN CONDENSATEUR ?

Il est composé de deux armatures conductrices séparées par un diélectrique isolant. Il stocke une charge électrique (Q) et la restitue lorsqu'il est connecté à un circuit.



Symbole :

- ✓ Stocke l'énergie
- ✓ Se charge et se décharge
- ✓ Bloque le courant continu (DC) après charge
- ✓ Laisse passer le courant alternatif (AC)

2 CAPACITÉ ET UNITÉS

La capacité (C) indique la quantité de charge stockée par volt.

$$C = \frac{Q}{V}$$

où :
Q : charge (Coulomb)
V : tension (Volt)

Unité : Farad (F)

En pratique, on utilise :

Unité	Symbole	Équivalent
Farad	F	1 F
Millifarad	mF	10^{-3} F
Microfarad	μF	10^{-6} F
Nanofarad	nF	10^{-9} F
Picofarad	pF	10^{-12} F

Les condensateurs courants sont en μF ou nF.

3 ÉNERGIE STOCKÉE

L'énergie (E) stockée dans un condensateur est donnée par :

$$E = \frac{1}{2} \times C \times V^2$$

où :
C : capacité (F)
V : tension (V)

Plus la capacité et la tension sont élevées, plus l'énergie stockée est grande.

EXEMPLE

Un condensateur de 1000 μF (0,001 F) chargé à 12 V stocke :
 $E = 0,5 \times 0,001 \times 12^2 = 0,072 \text{ J}$

4 TYPES DE CONDENSATEURS



Céramique (non polarisé)
Petites valeurs (pF à μF)
Haute fréquence, découplage



Électrolytique (polarisé)
Grandes valeurs (μF à mF)
Filtrage, alimentation



Film plastique (non polarisé)
Bonne stabilité, faible perte
Audio, temporisation



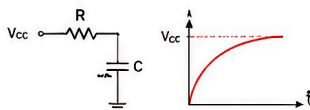
Tantale (polarisé)
Petite taille, bonne fiabilité
Électronique de précision



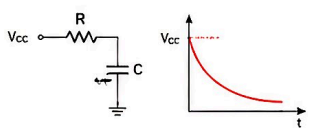
Respecter la polarité (+ et -) des condensateurs électrolytiques et au tantale.

5 CHARGE ET DÉCHARGE

Charge à travers une résistance (R)



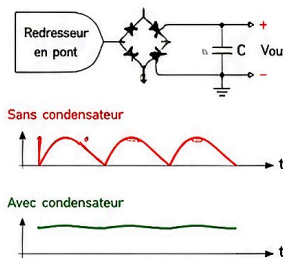
Décharge à travers une résistance (R)



La constante de temps $\tau = R \times C$ détermine la vitesse de charge/décharge.

6 CONDENSATEUR DE FILTRAGE

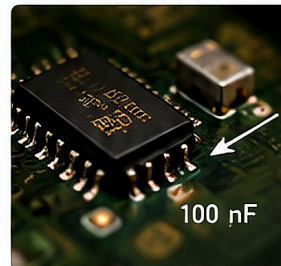
Après redressement, la tension contient des ondes (ripple). Le condensateur se charge au sommet et se décharge lentement pour lisser la tension.



✓ Réduit l'ondulation et fournit une tension DC plus stable.

7 DÉCOUPLAGE DES CIRCUITS

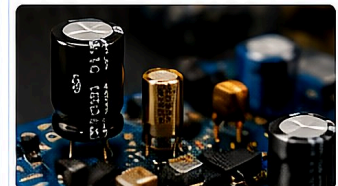
Les condensateurs céramiques de 100 nF sont utilisés près des circuits intégrés pour bloquer les parasites haute fréquence.



✓ Améliore la stabilité et empêche les parasites d'entrer dans le circuit.

8 APPLICATIONS EN ROBOTIQUE

- Lissage des alimentations DC
- Stockage d'énergie pour les pics de courant (moteurs, servos)
- Temporisation (circuits RC)
- Suppression du bruit et des parasites
- Stabilisation des capteurs sensibles



✓ Indispensable pour des systèmes robotiques fiables et performants.

9 VALEURS COURANTES



22 pF



100 nF
(0,1 μF)



1 μF
(non polarisé)



10 μF
16 V



100 μF
25 V



1000 μF
25 V

✓ Choisir la valeur et la tension nominale adaptées à votre circuit.

10 À RETENIR

- ✓ Le condensateur stocke de l'énergie dans un champ électrique.
- ✓ La capacité se mesure en Farad (F), souvent en μF ou nF.
- ✓ Il se charge et se décharge selon la constante de temps $\tau = R \times C$.
- ✓ Utile pour le filtrage, le découplage et la stabilisation des tensions.
- ✓ Respecter la polarité des condensateurs polarisés.

ASTUCE

Pour lisser efficacement une alimentation, utilisez une grande capacité ($\geq 100 \mu\text{F}$) et placez-la près de la charge.



2

CHAPITRE 2

Électronique
fondamentale

2.20

RÉSISTANCES ET DIVISEURS DE TENSION

Contrôler et partager la tension

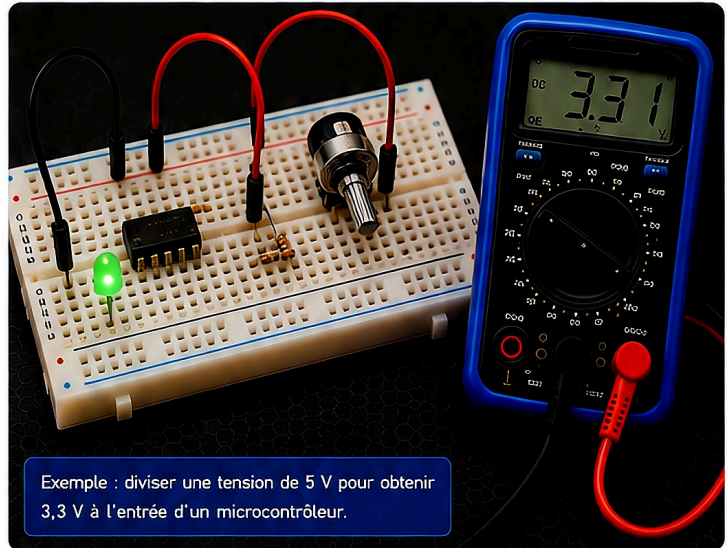
R



La résistance s'oppose au passage du courant électrique. Plus la résistance est grande, moins le courant passe.

Les résistances limitent le courant, protègent les composants et permettent de diviser la tension dans un circuit.

Le diviseur de tension est très utile pour adapter une tension élevée à un niveau plus faible, parfait pour les capteurs et les microcontrôleurs.



Exemple : diviser une tension de 5 V pour obtenir 3,3 V à l'entrée d'un microcontrôleur.

1 QU'EST-CE QU'UNE RÉSISTANCE ?

Une résistance limite le courant dans un circuit conformément à la loi d'Ohm.

$$V = R \times I$$

où :

V : tension (V)

R : résistance (Ω)

I : courant (A)

- ✓ Protège les composants sensibles
- ✓ Régule le courant
- ✓ Crée des chutes de tension
- ✓ Ajuste les signaux

2 CODE DES COULEURS

Les résistances sont marquées par des bandes de couleurs représentant leur valeur.



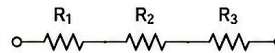
Couleur	Chiffre	Multiplieur	Tolérance
Noir	0	1 Ω	—
Marron	1	10 Ω	$\pm 1\%$
Rouge	2	100 Ω	$\pm 2\%$
Orange	3	1 k Ω	—
Jaune	4	10 k Ω	—
Vert	5	100 k Ω	$\pm 0,5\%$
Bleu	6	1 M Ω	$\pm 0,25\%$
Violet	7	10 M Ω	$\pm 0,10\%$
Gris	8	—	$\pm 0,05\%$
Blanc	9	—	—
Or	—	0,1	$\pm 5\%$
Argent	—	0,01	$\pm 10\%$

Exemple :
Rouge – Violet –
Marron – Or
 $= 27 \times 10^1$
 $= 270 \Omega \pm 5\%$

3 RÉSISTANCES EN SÉRIE

La résistance totale est la somme :

$$R_T = R_1 + R_2 + R_3 + \dots$$



Exemple :

$$R_1 = 220 \Omega, R_2 = 330 \Omega, R_3 = 470 \Omega$$

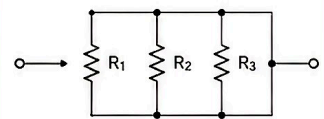
$$R_T = 220 + 330 + 470 = 1020 \Omega$$

En série, le courant est le même partout dans le circuit.

4 RÉSISTANCES EN PARALLÈLE

L'inverse de la résistance totale est la somme des inverses :

$$\frac{1}{R_T} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \dots$$



Exemple :

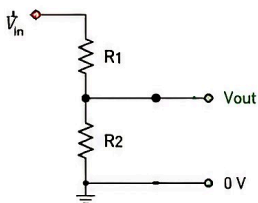
$$R_1 = 220 \Omega, R_2 = 330 \Omega, R_3 = 470 \Omega$$

$$R_T \approx 94,6 \Omega$$

En parallèle, la tension est la même aux bornes de chaque résistance.

5 DIVISEUR DE TENSION

Permet d'obtenir une tension plus faible à partir d'une tension plus élevée.



Formule :

$$V_{out} = V_{in} \times \frac{R_2}{R_1 + R_2}$$

6 EXEMPLE PRATIQUE

Diviser 5 V pour obtenir 3,3 V.

- Choisissons $R_1 = 1 \text{ k}\Omega$ et $R_2 = 2 \text{ k}\Omega$

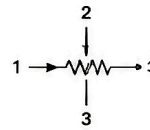
$$V_{out} = 5 \times \frac{2 \text{ k}\Omega}{1 \text{ k}\Omega + 2 \text{ k}\Omega}$$

$$V_{out} = 5 \times \frac{2}{3} = 3,33 \text{ V}$$

- ✓ Idéal pour adapter la tension d'un capteur à l'entrée d'un microcontrôleur (ex. 3,3 V).

7 POTENTIOMÈTRE

Une résistance variable (potentiomètre) permet d'ajuster la tension ou le courant.



Utilisations :

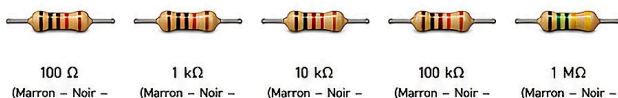
- Réglage de volume
- Ajustement de vitesse (moteurs)
- Calibration de capteurs

8 APPLICATIONS EN ROBOTIQUE

- Limiter le courant des LED et moteurs
- Créer des chutes de tension pour protéger les circuits
- Adapter les signaux des capteurs (capteurs analogiques)
- Ajuster la luminosité des écrans, des LED ou des affichages
- Commande de vitesse et de position des servomoteurs



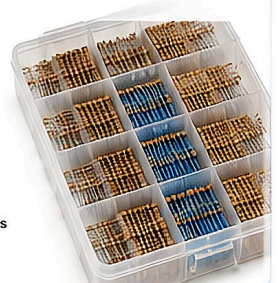
9 EXEMPLES DE RÉSISTANCES ET LEURS VALEURS COURANTES



Les résistances ont souvent une tolérance de $\pm 5\%$ (or) ou $\pm 10\%$ (argent).

10 À RETENIR

- ✓ La résistance limite le courant selon la loi d'Ohm.
- ✓ Les résistances peuvent être en série ou en parallèle.
- ✓ Le diviseur de tension permet de réduire une tension.
- ✓ Choisir la bonne valeur de résistance est essentiel pour la protection et le bon fonctionnement des circuits.
- ✓ Indispensable pour concevoir des systèmes robotiques fiables et sécurisés.



2

CHAPITRE 2 Électronique fondamentale

2.21

DIODES ET APPLICATIONS

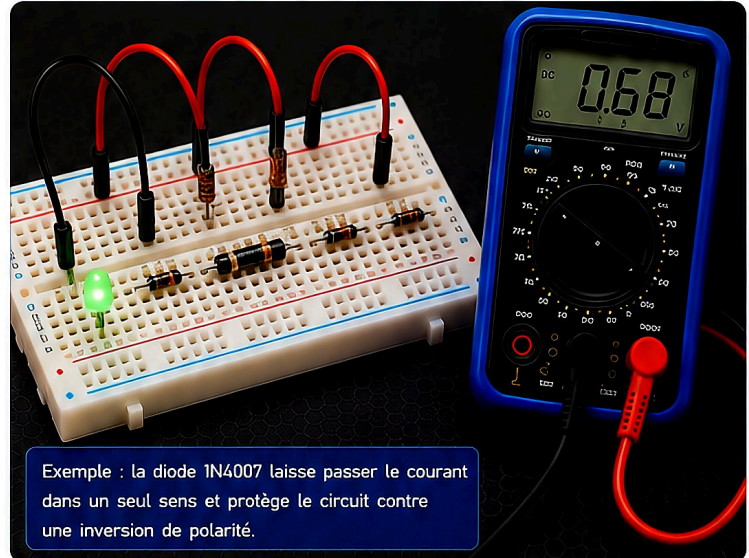
Contrôler le sens du courant



La diode laisse passer le courant dans un seul sens : de l'anode (A) vers la cathode (K). Elle protège les circuits contre les inversions de polarité.

La diode est un composant essentiel qui permet au courant de circuler dans une seule direction. Elle est largement utilisée pour la protection, le redressement et la commutation des signaux dans les systèmes robotiques.

Comprendre son fonctionnement et ses applications est indispensable pour concevoir des circuits fiables et sécurisés.



Exemple : la diode 1N4007 laisse passer le courant dans un seul sens et protège le circuit contre une inversion de polarité.

1 QU'EST-CE QU'UNE DIODE ?

Une diode est un composant semi-conducteur à deux bornes : anode (A) et cathode (K). Elle conduit le courant uniquement de A vers K.



- ✓ Conduit dans un seul sens
- ✓ Bloque dans l'autre sens
- ✓ Protège contre les inversions
- ✓ Utilisée en redressement et en commutation

2 SYMBOLE ET BROCHAGE

Symbole



Diode 1N4007 (exemple)



A (Anode)

K (Cathode)

La bande indique toujours la cathode (K). Le courant conventionnel circule de A vers K.

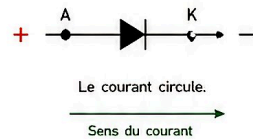
3 CARACTÉRISTIQUES PRINCIPALES

Tension inverse maximale (V_{RM})	1000 V
Courant direct moyen (I_F)	1 A
Tension directe (V_F)	$\approx 0,7$ V (Si)
Type	Redressement rapide
Boltier	DO-41

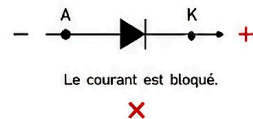
La tension directe typique d'une diode silicium est d'environ 0,6 – 0,7 V.

4 POLARISATION

Polarisation directe (conduction)

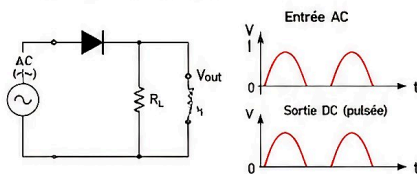


Polarisation inverse (blocage)



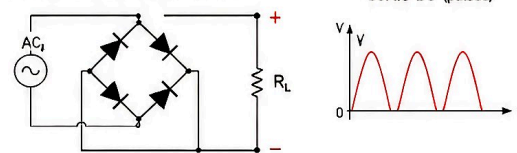
5 REDRESSEMENT DEMI-ONDE

La diode transforme une tension alternative (AC) en tension continue (DC) pulsée.



6 REDRESSEMENT PONT DE DIODES

Le pont de diodes permet d'obtenir une tension continue pulsée à partir d'une tension AC, en utilisant les deux alternances.



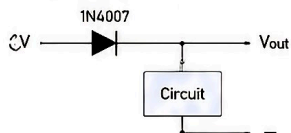
7 APPLICATIONS COURANTES

- Redressement de tension AC en DC
- Protection contre l'inversion de polarité
- Roue libre (freewheel) pour charges inductives
- Clamping et limitation de tension
- Indication de polarité (LED avec diode)
- Commutation rapide des signaux

8 EXEMPLES PRATIQUES

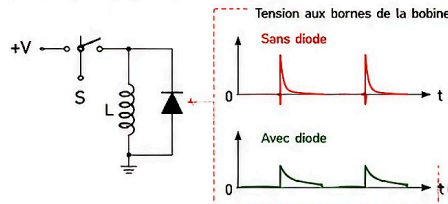
a) Protection contre l'inversion de polarité

La diode protège le circuit si la polarité de l'alimentation est inversée.



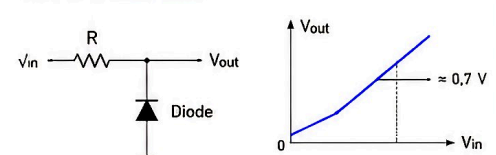
b) Diode de roue libre (sur charge inductive)

La diode absorbe la surtension générée par une bobine lors de l'arrêt du courant.



c) Limitation de tension (clamping)

La diode limite la tension à une valeur proche de sa tension directe.



9 TYPES DE DIODES

Diode redresseuse (1N400x)

Usage général pour redressement AC/DC.



Diode Schottky (1N5819)

Faible chute de tension, très rapide.



Diode Zener (BZX55C5V1)

Maintient une tension constante en inverse.



LED (diode électroluminescente)

Émet de la lumière quand elle est polarisée directement.



À RETENIR

- ✓ La diode laisse passer le courant dans un seul sens : de A vers K.
- ✓ En polarisation directe, elle conduit ; en inverse, elle bloque.
- ✓ Elle est essentielle pour le redressement, la protection et la commutation.
- ✓ Toujours respecter la polarité pour éviter d'endommager le circuit.
- ✓ Comprendre la diode est une base pour l'électronique et la robotique !

2

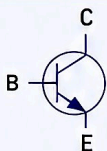
CHAPITRE 2

Électronique
fondamentale

2.22

TRANSISTORS BJT NPN ET PNP

Amplifier et commuter les signaux

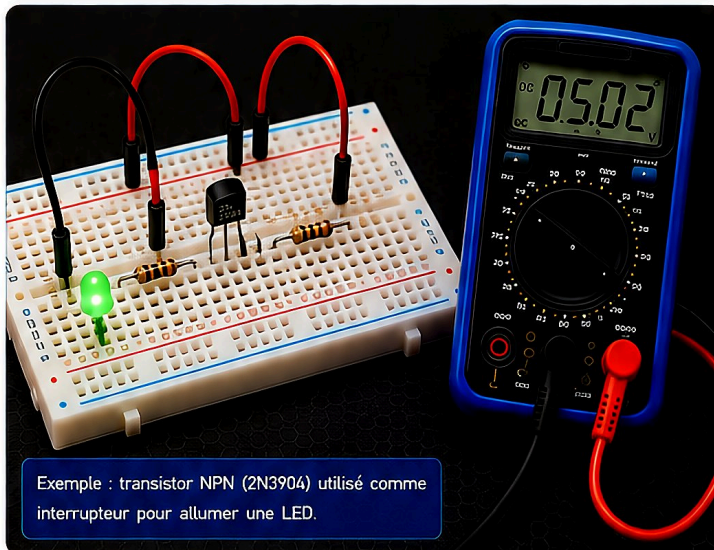


Le transistor (BJT) amplifie ou commute le courant à l'aide d'un petit signal appliqué à la base (B).

Le transistor bipolaire à jonction (BJT) est un composant actif à trois bornes : Base (B), Collecteur (C) et Émetteur (E).

Il peut fonctionner comme amplificateur de courant ou comme interrupteur électronique (ON/OFF).

Il existe deux types principaux : NPN et PNP, qui se différencient par le sens du courant et la polarité des tensions.



Exemple : transistor NPN (2N3904) utilisé comme interrupteur pour allumer une LED.

1 QU'EST-CE QU'UN TRANSISTOR ?

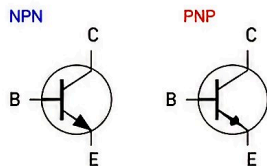
Un transistor BJT est constitué de trois régions dopées (N et P).

Un petit courant à la base contrôle un courant beaucoup plus grand entre le collecteur et l'émetteur.

Bornes	Fonction
B	Base
C	Collecteur
E	Émetteur

Le petit courant de base commande le grand courant C-E.

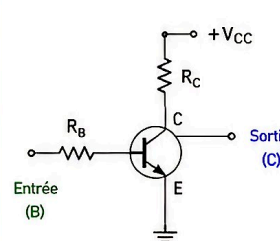
2 SYMBOLES DES TRANSISTORS



La flèche de l'émetteur indique le sens du courant conventionnel.

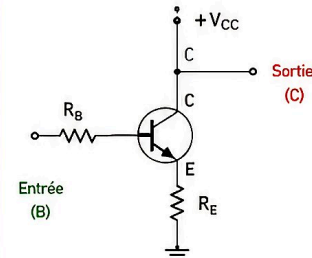
Type	Flèche émetteur	Courant principal
NPN	Sort (vers l'extérieur)	C → E
PNP	Entre (vers l'intérieur)	E → C

3 CONFIGURATION NPN (ÉMETTEUR COMMUN)



Le signal d'entrée est appliqué à la base. La sortie est prise au collecteur.

4 CONFIGURATION PNP (ÉMETTEUR COMMUN)



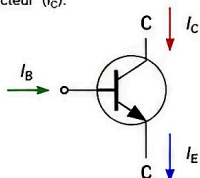
Le fonctionnement est similaire au NPN mais avec des polarités inversées.

5 CONDITIONS DE FONCTIONNEMENT (NPN)

Pour qu'un NPN conduise :

- Jonction B-E doit être polarisée en direct (~0,7 V pour le silicium)
- Jonction B-C doit être polarisée en inverse

Le courant de base (I_B) contrôle le courant collecteur (I_C).



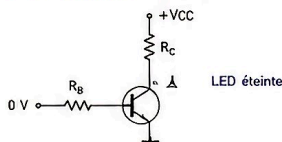
$$I_C \approx \beta \times I_B$$

β (gain en courant) typiquement 50 à 300

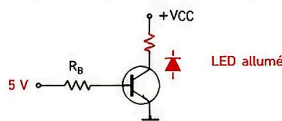
6 MODE COMMUTATEUR

Le transistor peut être utilisé comme interrupteur électronique.

- OFF (coupure) : $I_B = 0 \rightarrow I_C = 0$



- ON (saturation) : I_B suffisante $\rightarrow I_C$ max



✓ Idéal pour piloter des LED, relais, moteurs, etc.

7 EXEMPLE DE CALCUL

NPN 2N3904, $\beta = 100$

On veut $I_C = 50$ mA

Quel courant de base I_B minimal ?

Solution :

$$I_B = \frac{I_C}{\beta} = \frac{50 \text{ mA}}{100} = 0,5 \text{ mA}$$

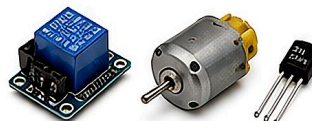
Si $V_{CC} = 5$ V et $R_B = 10$ k Ω :

$$I_B = \frac{V_{in} - V_{BE}}{R_B} = \frac{5 - 0,7}{10 \text{ k}\Omega} = 0,43 \text{ mA}$$

✓ I_B (0,43 mA) > 0,5 mA requis $\rightarrow$ Le transistor sature et la LED s'allume.

8 APPLICATIONS COURANTES

- Amplification de signaux faibles (audio, capteurs, microphones)
- Commutation de charges (LED, relais, moteurs)
- Oscillateurs et générateurs d'ondes
- Régulation et commande
- Interfaçage entre circuits logiques et puissance
- Circuits de protection et isolation optoélectronique



9 TYPES DE TRANSISTORS BJT COURANTS

Modèle	Type	I_C max	V_{CE} max	β (typ.)	Utilisation typique	Bollier	Illustration
2N3904	NPN	200 mA	40 V	100	Usage général, amplification	TO-92	
2N3906	PNP	200 mA	40 V	100	Complémentaire du 2N3904	TO-92	
BC547	NPN	100 mA	45 V	110	Amplification petit signal	TO-92	
TIP120	NPN	5 A	60 V	1000	Puissance moyenne, commande moteur	TO-220	
TIP127	PNP	5 A	60 V	1000	Complémentaire du TIP120	TO-220	

Toujours vérifier la fiche technique (datasheet) avant d'utiliser un transistor.

10 À RETENIR

- ✓ Le transistor BJT possède trois bornes : Base, Collecteur, Émetteur.
- ✓ Le NPN et le PNP diffèrent par le sens du courant et la polarité.
- ✓ Un petit courant de base contrôle un grand courant collecteur.
- ✓ Le transistor peut amplifier ou commuter (ON/OFF).
- ✓ En mode interrupteur, il doit être saturé (ON) ou coupé (OFF).
- ✓ Choisir le bon transistor et respecter ses limites de tension et de courant.
- ✓ Le transistor est un élément clé de l'électronique et de la robotique.



Conseil pratique

Pour tester un transistor NPN avec un multimètre :

- Pointe rouge sur B, noire sur E $\rightarrow$ ~0,6 à 0,7 V
- Pointe rouge sur C, noire sur B $\rightarrow$ ~0,6 à 0,7 V
- Inversement $\rightarrow$ OL (aucune conduction)



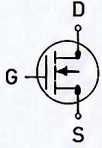
2

CHAPITRE 2 Électronique fondamentale

2.23

MOSFET ET APPLICATIONS

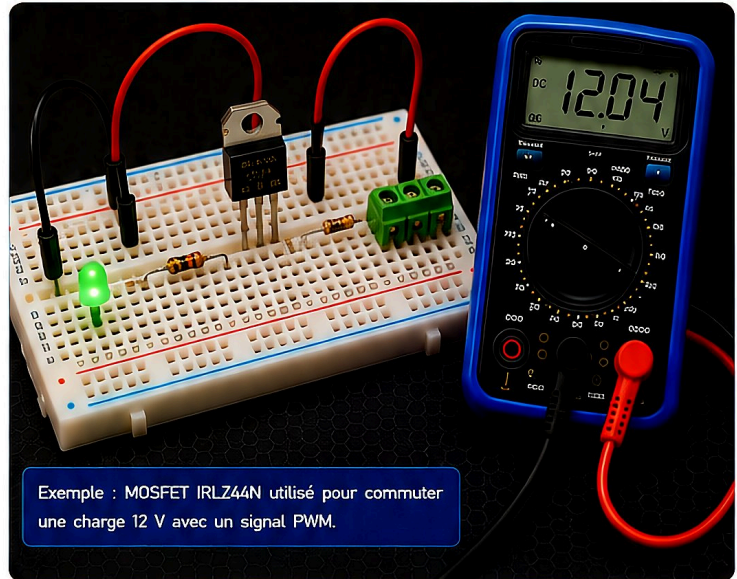
Contrôler la puissance avec efficacité



Le MOSFET (Metal-Oxide-Semiconductor Field-Effect Transistor) est un transistor commandé par la tension appliquée sur la grille (G). Il possède trois bornes : Grille (G), Drain (D) et Source (S).

Le MOSFET est un transistor à effet de champ qui offre une très haute impédance d'entrée, une faible résistance à l'état passant et une excellente efficacité pour la commutation rapide.

Il est largement utilisé dans le contrôle de moteurs, les alimentations à découpage, les onduleurs et les circuits de puissance en robotique.



Exemple : MOSFET IRLZ44N utilisé pour commuter une charge 12 V avec un signal PWM.

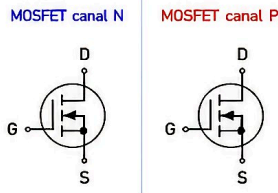
1 QU'EST-CE QU'UN MOSFET ?

Le MOSFET est un transistor unipolaire à effet de champ. Le courant entre le Drain (D) et la Source (S) est contrôlé par la tension appliquée sur la Grille (G).

Bornes	Fonction
G Grille	Commande (tension)
D Drain	Courant de sortie
S Source	Référence (0 V)

- ✓ Très faible courant d'entrée
- ✓ Idéal pour la commutation rapide
- ✓ Faible résistance à l'état passant
- ✓ Haute efficacité énergétique

2 TYPES DE MOSFET

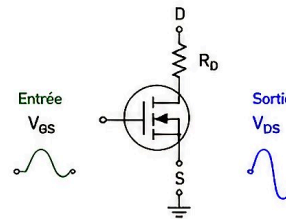


Type	Courant	Commande
Canal N	D → S	V_{GS} positif
Canal P	S → D	V_{GS} négatif

Les MOSFET canal N sont les plus utilisés en électronique de puissance.

3 MODE D'AMPLIFICATION (LINÉAIRE)

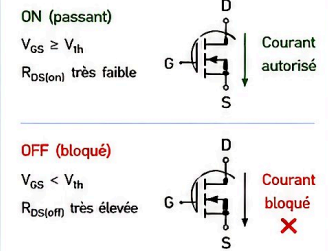
Le MOSFET fonctionne comme un amplificateur quand il n'est pas complètement ouvert.



Utilisé dans les circuits analogiques, amplificateurs audio et linéaires.

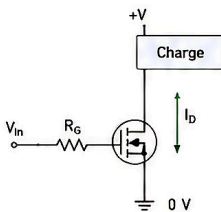
4 MODE COMMUTATEUR (ON/OFF)

Le MOSFET est le plus souvent utilisé en commutation : totalement ouvert (ON) ou fermé (OFF).



Idéal pour contrôler des charges importantes avec un faible signal.

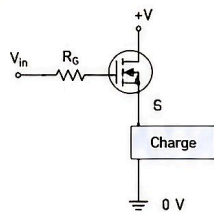
5 CIRCUIT DE COMMANDE BASSE CÔTÉ



La charge est connectée entre +V et le Drain. Le MOSFET commute à la masse (côté bas).

✓ Configuration la plus courante et simple.

6 CIRCUIT DE COMMANDE HAUTE CÔTÉ

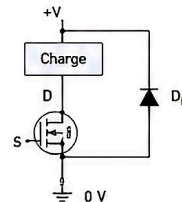


La charge est connectée entre le Drain et la masse. Le MOSFET commute à +V.

✓ Nécessite un pilotage spécial (driver) pour élever la tension de grille.

7 DIODE DE ROUE LIBRE

Utilisée avec les charges inductives (moteurs, relais, solénoïdes) pour protéger le MOSFET contre les surtensions.



✓ La diode absorbe l'énergie de retour lorsque le MOSFET se ferme.

8 APPLICATIONS COURANTES

- Commande de moteurs DC et pas à pas
- Alimentation à découpage (SMPS)
- Onduleurs et convertisseurs DC-DC
- Variateurs de vitesse (ESC)
- Contrôle d'éclairage LED haute puissance
- Protection électronique et gestion de batterie



9 EXEMPLES DE MOSFET COURANTS

Modèle	Type	V_{DS} max	I_D max	$R_{DS(on)}$ typ.	Bolter	Utilisation typique	Image
IRLZ44N	Canal N	55 V	47 A	0,022 Ω	TO-220	Commutation générale	
IRF540N	Canal N	100 V	33 A	0,077 Ω	TO-220	Alimentations, moteurs	
IRF3205	Canal N	55 V	110 A	0,008 Ω	TO-220	Haute performance	
FQP30N06L	Canal N	60 V	30 A	0,035 Ω	TO-220	DC-DC, SMPS	

✓ Toujours vérifier la datasheet avant d'utiliser un MOSFET dans un circuit.

10 À RETENIR

- ✓ Le MOSFET est commandé par la tension de grille.
- ✓ Il possède une très faible résistance à l'état passant.
- ✓ Deux types principaux : canal N (le plus courant) et canal P.
- ✓ Utilisé en mode linéaire pour l'amplification ou en mode ON/OFF pour la commutation.
- ✓ Toujours protéger les charges inductives avec une diode de roue libre.
- ✓ Choisir un MOSFET selon la tension, le courant et la résistance à l'état passant.
- ✓ Le MOSFET est la clé de nombreux circuits de puissance modernes.



Conseil pratique

Un MOSFET chauffe ! Utilisez toujours un dissipateur thermique si le courant est élevé.



2

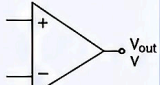
CHAPITRE 2

Électronique
fondamentale

2.24

AMPLIFICATEURS OPÉRATIONNELLS

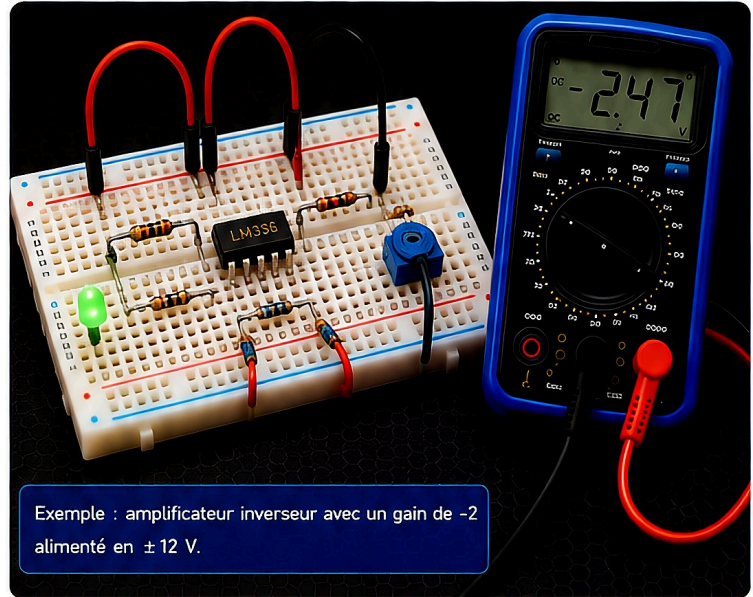
Le cœur du traitement analogique



L'amplificateur opérationnel (op-amp) est un amplificateur différentiel à très gain élevé. Il permet d'effectuer des opérations analogiques avec des composants externes simples.

L'amplificateur opérationnel, ou op-amp, est un circuit intégré très polyvalent. Avec seulement quelques résistances et condensateurs, il permet de réaliser une grande variété de fonctions analogiques : amplification, filtrage, sommation, intégration, etc.

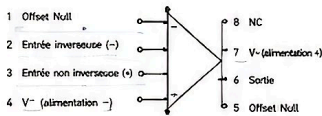
Sa compréhension est essentielle pour tout ingénieur en électronique et en robotique.



Exemple : amplificateur inverseur avec un gain de -2 alimenté en ± 12 V.

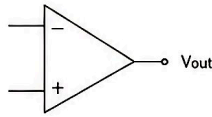
1 QU'EST-CE QU'UN OP-AMP ?

Un op-amp est un amplificateur différentiel à très haut gain. Il possède généralement 5 broches (pour un DIP-8) :



- ✓ Gain en boucle ouverte très élevé (10^2 à 10^6)
- ✓ Impédance d'entrée très élevée
- ✓ Impédance de sortie très faible
- ✓ Large bande passante

2 SYMBOLE ET NOTATIONS



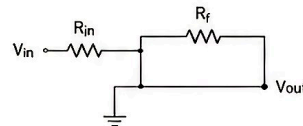
V^- : Entrée inverseuse
 V^+ : Entrée non inverseuse
 V_{out} : Sortie

Hypothèses idéales (rétroaction négative)

- ✓ $V^+ \approx V^-$ (masse virtuelle si $V^+ = 0$)
- ✓ Aucun courant n'entre dans les entrées
- ✓ Gain en boucle ouverte $A \rightarrow \infty$

3 AMPLIFICATEUR INVERSEUR

Le signal est appliqué sur l'entrée (-). L'entrée (+) est à la masse.



$$A_v = \frac{V_{out}}{V_{in}} = -\frac{R_f}{R_{in}}$$

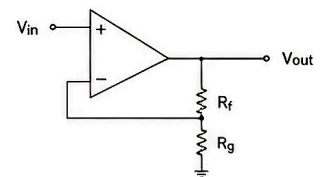
Exemple :

Si $R_{in} = 10 \text{ k}\Omega$ et $R_f = 20 \text{ k}\Omega$

$$A_v = -\frac{20k}{10k} = -2$$

4 AMPLIFICATEUR NON-INVERSEUR

Le signal est appliqué sur l'entrée (+).



$$A_v = \frac{V_{out}}{V_{in}} = 1 + \frac{R_f}{R_g}$$

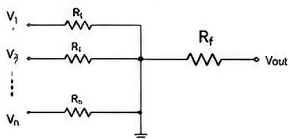
Exemple :

Si $R_f = 20 \text{ k}\Omega$ et $R_g = 10 \text{ k}\Omega$

$$A_v = 1 + \frac{20k}{10k} = 3$$

5 SOMMATEUR INVERSEUR

Permet de sommer plusieurs signaux.



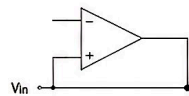
$$V_{out} = -R_f \left(\frac{V_1}{R_1} + \frac{V_2}{R_2} + \dots + \frac{V_n}{R_n} \right)$$



Très utile pour le mixage audio, le traitement de capteurs, etc.

6 SUIVEUR DE TENSION (BUFFER)

Sortie reliée directement à l'entrée (-).

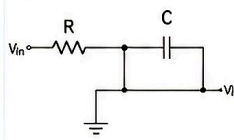


$$A_v = 1$$

$$V_{out} = V_{in}$$

- ✓ Isolé la source de la charge (adaptation d'impédance).

7 INTÉGRATEUR

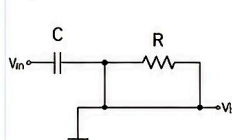


$$V_{out} = -\frac{1}{RC} \int V_{in} dt$$

Utilisations :

- Générateurs de rampe
- Filtrage actif
- Conversion A/N simple

8 DIFFÉRENCIEUR

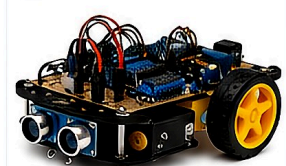


$$V_{out} = -RC \frac{dV_{in}}{dt}$$

Utilisations :

- Détection de fronts
- Filtrage passe-haut
- Circuit de temporisation

9 APPLICATIONS EN ROBOTIQUE



- Conditionnement des signaux des capteurs
- Amplification de faibles tensions (thermistors, photorésistances)
- Filtrage du bruit analogique
- Commande de moteurs et servos
- Génération de signaux et formes d'onde

10 CARACTÉRISTIQUES COURANTES

Paramètre	Typique (LM358)	Description
Tension d'alimentation	$\pm 3 \text{ V}$ à $\pm 16 \text{ V}$	Plage d'alimentation
Gain en boucle ouverte (A_{OL})	100 dB (10^5)	Très élevé
Impédance d'entrée	$\sim 10^9 \Omega$	Très grande
Impédance de sortie	$\sim 50 \Omega$	Très faible
Bande passante (GBW)	1 MHz	Gain $\times$ Bande passante
Courant de sortie max	$\pm 20 \text{ mA}$	Capacité de courant



Les valeurs varient selon le modèle (LM324, TL082, TL081, etc.).

11 CONSEILS DE CONCEPTION

- ✓ Toujours utiliser une rétroaction négative.
- ✓ Ajouter des résistances de compensation si nécessaire pour éviter les oscillations.
- ✓ Respecter les plages de tension d'entrée et de sortie.
- ✓ Découpler l'alimentation avec 100 nF près du circuit intégré.
- ✓ Utiliser des résistances de précision pour des résultats stables.

12 À RETENIR

- ✓ L'op-amp est un amplificateur différentiel très polyvalent.
- ✓ Les montages de base : inverseur, non-inverseur, suiveur.
- ✓ Il permet de réaliser des fonctions mathématiques : somme, intégration, dérivation, filtrage.
- ✓ Outil indispensable en électronique analogique et en robotique !



Astuce

Commencez toujours vos essais avec un suiveur de tension pour vérifier le fonctionnement de votre circuit et de votre alimentation.



2

CHAPITRE 2

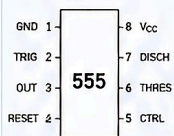
Électronique
fondamentale

2.25

CIRCUIT INTÉGRÉ NE555 ET APPLICATIONS

Le temporisateur universel

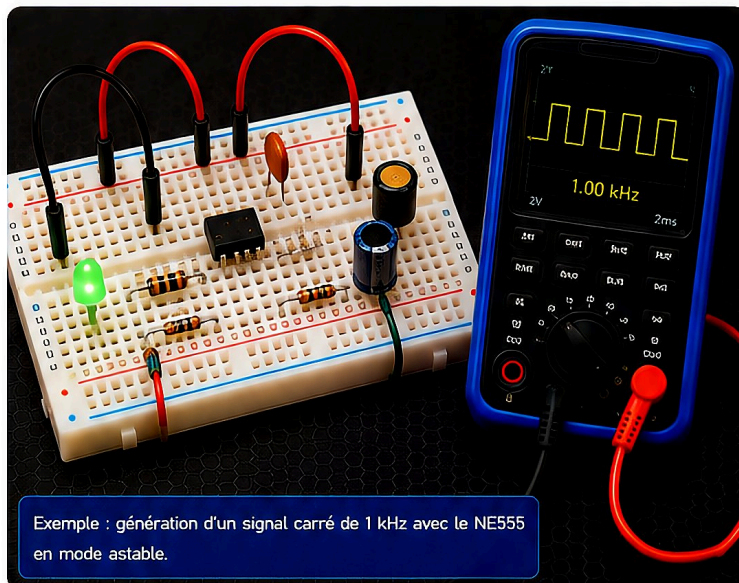
Symbole du NE555



Le NE555 est un circuit intégré très polyvalent utilisé comme temporisateur, oscillateur ou générateur d'impulsions.

Le NE555 est l'un des circuits intégrés les plus utilisés en électronique. Il peut fonctionner en mode monostable, astable ou bistable pour générer des temporisations précises ou des signaux périodiques.

Il est idéal pour les minuteries, les clignotants LED, les générateurs d'ondes carrées et de nombreuses applications en robotique.



Exemple : génération d'un signal carré de 1 kHz avec le NE555 en mode astable.

1 PRÉSENTATION DU NE555

Le NE555 est composé de comparateurs, d'un bascule RS, d'un transistor de décharge et d'un étage de sortie.

- Grande précision
- Tension d'alimentation : 4,5 V à 16 V
- Courant de sortie jusqu'à 200 mA
- Fonctionne en 3 modes principaux



Très utilisé dans les systèmes électroniques et robotiques.

2 BROCHAGE DU NE555

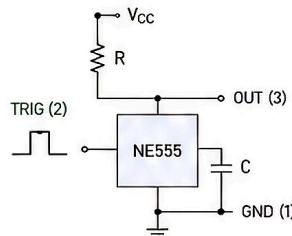
1	GND	Masse (0 V)
2	TRIG	Déclenchement
3	OUT	Sortie
4	RESET	Remise à zéro
5	CTRL	Tension de contrôle
6	THRES	Seuil (threshold)
7	DISCH	Décharge
8	VCC	Alimentation (+V)



Toujours découpler la broche 5 avec un condensateur de 10 nF vers la masse.

3 MODE MONOSTABLE

Produit une seule impulsion à la réception d'un déclenchement négatif (broche 2).

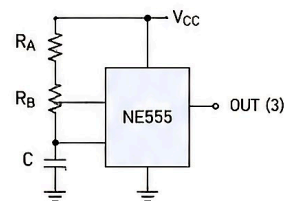


$$\text{Durée de l'impulsion : } T = 1,1 R C$$

Utilisation : minuterie, temporisation.

4 MODE ASTABLE

Génère un signal carré continu (oscillateur).



$$T_{ON} = 0,693 (R_A + R_B) C$$

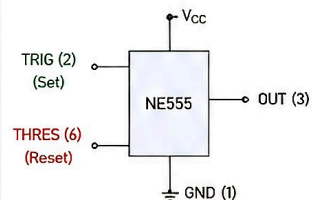
$$T_{OFF} = 0,693 R_B C$$

$$f = \frac{1,44}{(R_A + 2R_B) C}$$

Utilisation : clignotants, PWM, horloges.

5 MODE BISTABLE

Fonctionne comme un bascule (mémoire). L'état change selon les impulsions sur TRIG (Set) et THRES (Reset).



Utilisation : mémoires, verrous électroniques, commande marche/arrêt.

6 EXEMPLE PRATIQUE – ASTABLE

Concevoir un oscillateur de 2 Hz avec :

$$R_A = 1 \text{ k}\Omega, R_B = 10 \text{ k}\Omega$$

Calcul :

$$f = \frac{1,44}{(R_A + 2R_B) C}$$

$$2 = \frac{1,44}{(1k + 2 \times 10k) C}$$

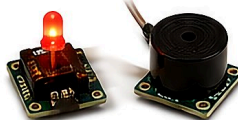
$$C = \frac{1,44}{2 \times 21k} \approx 34,3 \mu\text{F}$$

Choisir $C = 33 \mu\text{F}$ (valeur standard)

- ✓ Le signal de sortie est une onde carrée de 2 Hz (50% environ).

7 APPLICATIONS COURANTES

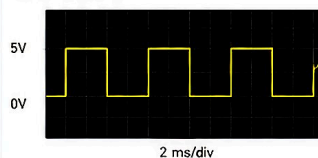
- Minuteries et temporisateurs
- Générateurs d'ondes carrées
- PWM pour contrôle de moteur
- Clignotants LED
- Alarme et sirènes
- Retard à l'enclenchement
- Détecteurs et capteurs de temps



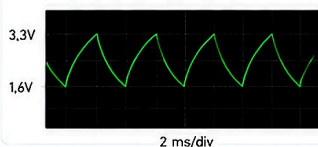
8 FORMES D'ONDE – ASTABLE

Signaux mesurés à l'oscilloscope.

Sortie (broche 3)

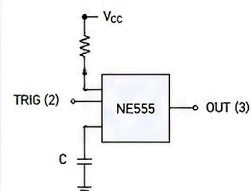


Tension sur le condensateur (broche 6)



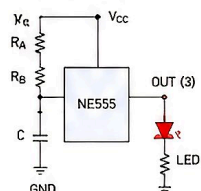
9 EXEMPLES DE CIRCUITS AVEC NE555

a) Monostable – Retard à l'enclenchement



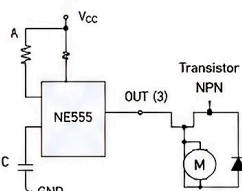
Sortie : impulsion unique

b) Astable – Clignotant LED



Sortie : clignotement continu

c) PWM pour moteur DC



Commande de vitesse par PWM



Toujours utiliser un condensateur de découplage (100 nF) entre VCC (broche 8) et GND (broche 1) le plus près possible du circuit.

10 À RETENIR

- ✓ Le NE555 est un circuit intégré très polyvalent.
- ✓ Il peut fonctionner en trois modes : monostable, astable et bistable.
- ✓ En mode astable, il génère un signal carré continu dont la fréquence dépend de R_A , R_B et C .
- ✓ En mode monostable, il produit une impulsion de durée $T = 1,1 R C$.
- ✓ Il est indispensable dans de nombreuses applications électroniques et robotiques.



Conseil pratique

Pour de bonnes performances, utilisez des résistances de précision et des condensateurs de qualité. Testez toujours le circuit à l'oscilloscope !



2

CHAPITRE 2

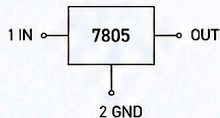
Électronique
fondamentale

2.26

RÉGULATEURS DE TENSION ET ALIMENTATIONS

Fournir une tension stable et fiable

Symbole d'un régulateur
(ex. 7805)



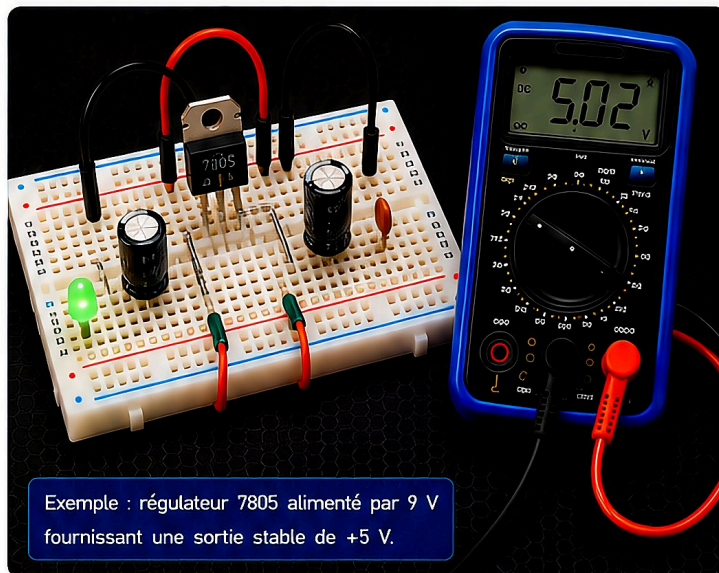
IN : Entrée non régulée (V_{in})

OUT : Sortie régulée (V_{out})

GND : Masse (0 V)

Les régulateurs de tension délivrent une tension de sortie constante, indépendante des variations de la tension d'entrée ou de la charge.

Ils protègent les circuits sensibles et assurent un fonctionnement stable dans les systèmes électroniques et robotiques.



Exemple : régulateur 7805 alimenté par 9 V fournissant une sortie stable de +5 V.

1 POURQUOI RÉGULER LA TENSION ?

Les circuits électroniques nécessitent une tension stable pour fonctionner correctement.

Sans régulation :

- ✗ Les performances varient.
- ✗ Les composants peuvent être endommagés.
- ✗ Le bruit et les interférences augmentent.

Avec régulation :

- ✓ Tension constante et propre
- ✓ Protection des composants
- ✓ Meilleure fiabilité du système

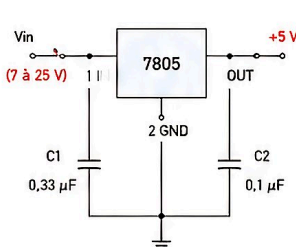
Toujours réguler la tension avant d'alimenter des circuits sensibles.

2 RÉGULATEURS LINÉAIRES POPULAIRES

Modèle	V_{out} fixe	Courant max	Boîtier
7805	+5 V	1,5 A	TO-220
7812	+12 V	1,5 A	TO-220
7905	-5 V	1,5 A	TO-220
LM317	1,25 à 37 V (ajustable)	1,5 A	TO-220

Les 78xx sont des régulateurs positifs, les 79xx sont négatifs. Le LM317 est ajustable.

3 CIRCUIT DE BASE (RÉGULATEUR 7805)

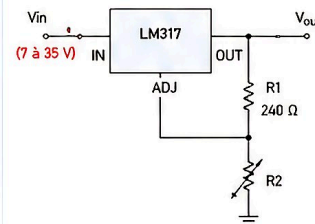


C1 : Filtre l'entrée

C2 : Améliore la stabilité de la sortie

Ces condensateurs sont indispensables pour éviter les oscillations.

4 RÉGULATEUR AJUSTABLE (LM317)



$$V_{out} = 1,25 \left(1 + \frac{R_2}{R_1} \right) + I_{adj} R_2$$

$I_{adj} \approx 50 \mu A$ (peut être négligé)

Idéal pour obtenir une tension variable et personnalisée.

5 DISSIPATION DE PUISSANCE

Un régulateur linéaire dissipe la différence de tension sous forme de chaleur.

Formule :

$$P_D = (V_{in} - V_{out}) \times I_{out}$$

Exemple : $V_{in} = 12 V$, $V_{out} = 5 V$, $I_{out} = 0,5 A$

$$P_D = (12 - 5) \times 0,5 = 3,5 W$$

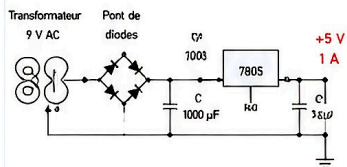
Utiliser un dissipateur thermique si P_D est supérieur à ~1 W.

Plus la différence de tension est grande, plus la chaleur à dissiper est importante.



6 ALIMENTATION LINÉAIRE COMPLETE

Exemple : alimentation 5 V / 1 A



Découplage recommandé :

- C1 (entrée) : 0,33 µF
- C2 (sortie) : 0,1 µF

Assure une tension continue filtrée et régulée.

7 ALIMENTATION À DÉCOUPAGE (SMPS)

Plus efficace pour les applications à forte puissance.

Avantages :

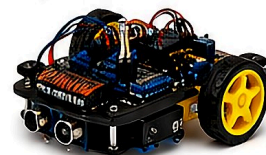
- Efficacité élevée (85–95 %)
- Moins d'échauffement
- Compact et léger
- Large plage de tension d'entrée



Idéal pour les moteurs, LED de puissance, cartes de commande et robots.

8 APPLICATIONS COURANTES

- Alimentation de microcontrôleurs (Arduino, ESP32, PIC, etc.)
- Alimentation de capteurs et modules
- Alimentation d'amplificateurs audio
- Alimentation de moteurs (via contrôleurs)
- Systèmes embarqués et robots
- Chargeurs de batterie



9 VALEURS DE CONDENSATEURS RECOMMANDÉES

Position	Valeur typique	Type conseillé	Rôle
Entrée (C1)	0,33 µF à 1 µF	Céramique ou film	Filtre les pics et bruit d'entrée
Sortie (C2)	0,1 µF	Céramique	Stabilité haute fréquence
Sortie (C3)	10 µF à 100 µF	Électrolytique	Réserve d'énergie et filtrage

Placer les condensateurs le plus près possible des broches du régulateur.

10 À RETENIR

- ✓ Un régulateur fournit une tension de sortie constante malgré les variations.
- ✓ Utiliser les condensateurs recommandés pour garantir la stabilité.
- ✓ Vérifier la dissipation thermique et utiliser un dissipateur si nécessaire.
- ✓ Le LM317 permet d'obtenir une tension ajustable facilement.
- ✓ Pour les fortes puissances, préférer une alimentation à découpage (SMPS).
- ✓ Une bonne alimentation = un système fiable et durable en robotique !

Conseil pratique

Toujours mesurer la tension de sortie avec un multimètre avant de connecter vos circuits sensibles ou votre robot.



2

CHAPITRE 2

Électronique
fondamentale

2.27

DIODES ET APPLICATIONS

Le courant dans un seul sens

Symbole d'une diode



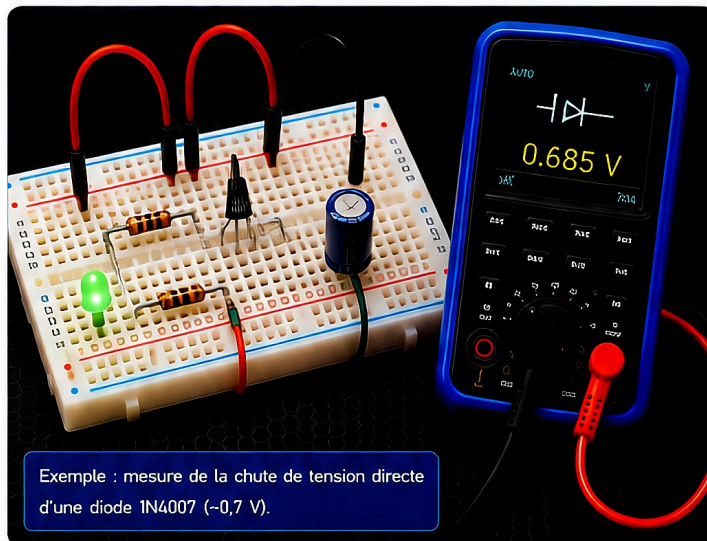
A

K

La diode laisse passer le courant de l'anode (A) vers la cathode (K), mais le bloque dans le sens inverse.

La diode est un composant à semi-conducteur qui ne laisse passer le courant électrique que dans un seul sens.

Elle est largement utilisée pour le redressement, la protection, la détection et la régulation de circuits.



Exemple : mesure de la chute de tension directe d'une diode 1N4007 (~0,7 V).

1 QU'EST-CE QU'UNE DIODE ?

Une diode est un composant à deux bornes (A et K) qui permet le passage du courant dans un seul sens.

Caractéristique	Description
Conduction directe	A → K (tension positive sur l'anode)
Blocage inverse	K → A (tension inversée)
Chute de tension directe (V_F)	Environ 0,6 – 0,7 V (diodes silicium)
Courant inverse (I_R)	Très faible (μA)
Utilisation principale	Redressement, protection, détection, régulation

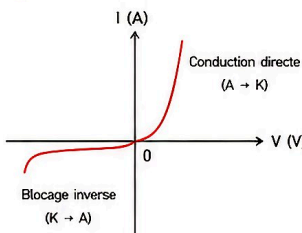
Toujours respecter la polarité de la diode dans un circuit.

2 TYPES DE DIODES COURANTES

Type	Symbole	Usage principal
Diode de redressement (1N4007)		Redressement AC → DC
Diode Zener (BZX55C5V1)		Régulation de tension
LED (DEL)		Indication lumineuse
Diode de signal (1N4148)		Commutation rapide et détection
Diode Schottky (1N5819)		Faible chute de tension, haute efficacité

Chaque diode a une application spécifique.

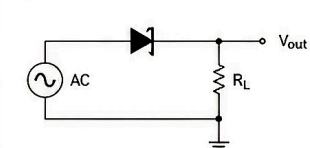
3 CARACTÉRISTIQUE I-V D'UNE DIODE



En conduction directe, la diode commence à conduire à partir d'environ 0,6 – 0,7 V (silicium).

Dans le sens inverse, le courant est quasi nul jusqu'à une très haute tension.

4 DIODE DE REDRESSEMENT DEMI-ONDE



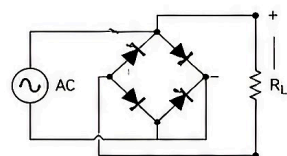
La diode ne laisse passer que les alternances positives.

Tension moyenne en sortie (idéale) :

$$V_{DC} = \frac{V_m}{\pi}$$

Simple, économique, mais la sortie contient beaucoup d'ondulation.

5 REDRESSEMENT DOUBLE ALTERNANCE



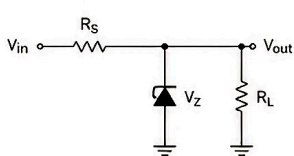
Les deux alternances sont utilisées, ce qui réduit l'ondulation et augmente la tension moyenne.

Tension moyenne (idéale) :

$$V_{DC} = \frac{2V_m}{\pi}$$

Nécessite 4 diodes, meilleure efficacité et meilleure utilisation du transformateur.

6 DIODE ZENER – RÉGULATION



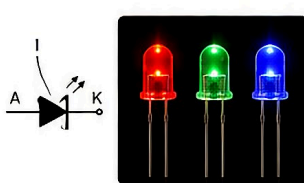
La diode Zener maintient une tension constante V_Z sur la charge R_L .

$$V_{out} \approx V_Z$$

si I_Z est dans la plage spécifiée.

Choisir R_S pour que le courant dans la Zener reste entre $I_{Z(min)}$ et $I_{Z(max)}$.

7 LED – DIODE ÉLECTROLUMINESCENTE



La LED émet de la lumière lorsque le courant passe de l'anode vers la cathode.

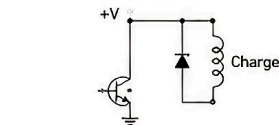
V_F typique :

- Rouge : 1,8 – 2,2 V
- Vert : 2,0 – 2,4 V
- Bleu : 3,0 – 3,3 V

Toujours utiliser une résistance en série pour limiter le courant.

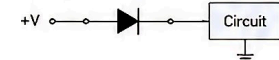
8 PROTECTION PAR DIODE

a) Diode de roue libre (flyback)



Protège le transistor contre les pics de tension induits par une charge inductive.

b) Protection d'inversion de polarité



Empêche la polarité inversée d'endommager le circuit.

Les diodes de protection augmentent la fiabilité des circuits.

9 VALEURS TYPIQUES DE DIODES COURANTES

Modèle	Type	V_F (typ.)	I_F max	V_R max	Usage typique	Boîtier
1N4007	Redressement	0,7 V	1 A	1000 V	Redressement AC-DC	DO-41
1N4148	Signal	0,7 V	150 mA	100 V	Commutation rapide	DO-35
1N5819	Schottky	0,3 – 0,4 V	1 A	40 V	Alimentations à haute efficacité	DO-41
BZX55C5V1	Zener	–	500 mW	5,1 V	Régulation 5,1 V	DO-35
LED Rouge	LED	1,8 – 2,2 V	20 mA	5 V	Indication	5 mm

Toujours vérifier la datasheet du composant avant utilisation.

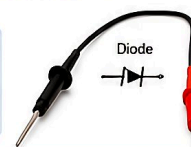
10 À RETENIR

- ✓ La diode laisse passer le courant uniquement de l'anode vers la cathode.
- ✓ Elle est essentielle pour le redressement, la protection et la régulation.
- ✓ Respecter la polarité lors du montage.
- ✓ Utiliser une résistance de limitation avec les LED.
- ✓ Les diodes Zener maintiennent une tension constante.
- ✓ Les diodes de roue libre protègent les circuits inductifs.
- ✓ Les diodes Schottky offrent une faible chute de tension et une haute efficacité.



Conseil pratique

Utilisez un multimètre en mode « diode » pour vérifier une diode : environ 0,6 – 0,7 V en sens direct, et OL (aucune conduction) en sens inverse.



2

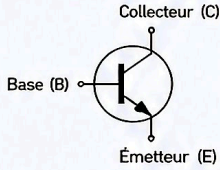
CHAPITRE 2 Électronique fondamentale

2.28

TRANSISTORS BJT ET APPLICATIONS

L'amplification et la commutation

Symbole et broches (NPN)

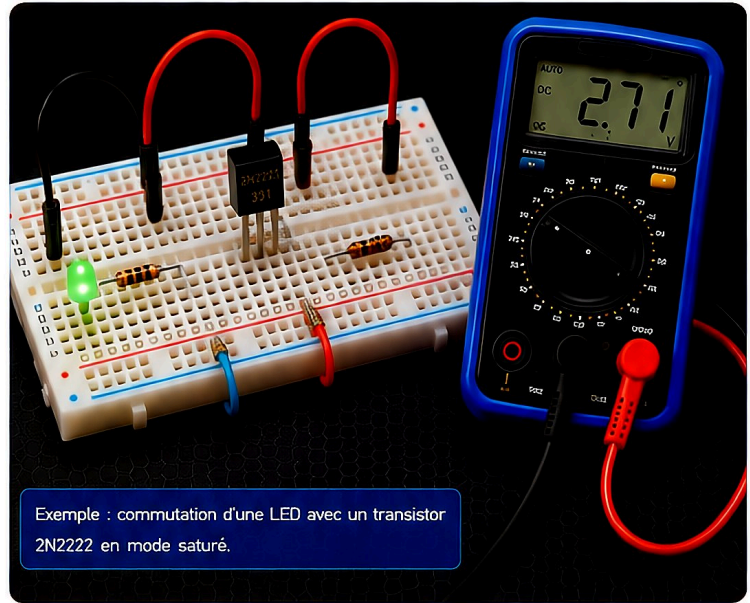


Le transistor BJT possède trois bornes : Base (B), Collecteur (C) et Émetteur (E). Le courant de base contrôle le courant entre le collecteur et l'émetteur.

Le transistor bipolaire à jonction (BJT) est un composant actif capable d'amplifier ou de commuter des signaux électriques.

Un faible courant appliqué à la base (B) contrôle un courant beaucoup plus important entre le collecteur (C) et l'émetteur (E).

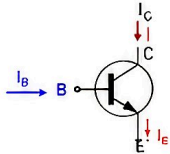
Les transistors sont essentiels dans les amplificateurs, les circuits logiques et les systèmes de commande.



Exemple : commutation d'une LED avec un transistor 2N2222 en mode saturé.

1 FONCTIONNEMENT DE BASE

Le transistor agit comme un robinet contrôlé.
– Un petit courant de base (I_B) permet un grand courant Collecteur-Émetteur (I_C).



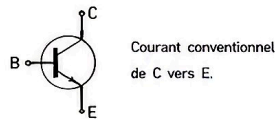
$$I_C \approx \beta \times I_B$$

β : gain en courant (h_{FE})

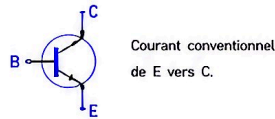
Plus le gain β est élevé, plus le transistor peut amplifier un signal.

2 CONFIGURATIONS BJT

NPN (le plus courant)



PNP



NPN et PNP sont complémentaires dans les circuits.

3 RÉGIONS DE FONCTIONNEMENT

Coupeure (OFF)

$$I_B = 0 \rightarrow I_C \approx 0$$

Le transistor est ouvert.

Active (Amplification)

$$0 < I_B < I_{B(sat)}$$

$$I_C = \beta \times I_B$$

Le transistor amplifie.

Saturation (ON)

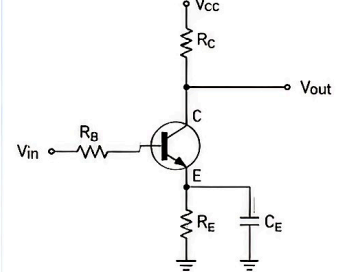
$$I_B \geq I_{B(sat)}$$

$$V_{CE} \approx 0.1 \text{ à } 0.3 \text{ V}$$

Le transistor est fermé.

En commutation, on utilise les zones Coupeure et Saturation.

4 MONTAGE AMPLIFICATEUR ÉMETTEUR COMMUN

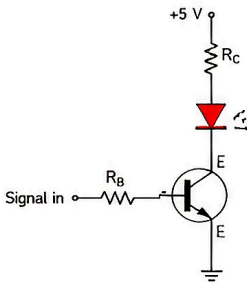


Ce montage inverse le signal et offre un bon gain en tension.

Idéal pour les amplificateurs audio et les préamplificateurs.

5 COMMUTATION AVEC BJT (NPN)

Exemple : allumer une LED



Lorsque le signal est haut, la LED s'allume. Sinon, elle reste éteinte.

6 CALCUL DES RÉSISTANCES (COMMUTATION)

a) Résistance de base (R_B)

$$R_B = \frac{V_{in} - V_{BE}}{I_B}$$

$$V_{BE} \approx 0.7 \text{ V (Silicium)}$$

b) Courant de base minimal

$$I_B \geq \frac{I_C}{\beta_{forcé}}$$

$$(\beta_{forcé} = 5 \text{ à } 10 \text{ en saturation})$$

Forcer le gain (β) assure une saturation fiable du transistor.

7 TRANSISTORS COURANTS

Modèle	Type	I_C max	V_{CE} max	Gain (β) typ.
2N2222	NPN	600 mA	40 V	100
BCS47	NPN	100 mA	45 V	200
TIP31C	NPN	3 A	100 V	15
2N2907	PNP	600 mA	60 V	100



Choisir le transistor selon la tension, le courant et le gain requis.

8 APPLICATIONS COURANTES

- Amplification de signaux audio
- Commandes de moteurs DC et relais
- Générateurs d'oscillations
- Circuits logiques (TTL, CMOS pilotes)
- Régulation et stabilisation
- Interfaces entre microcontrôleurs et charges puissantes
- Détection et conditionnement de signaux



9 RÉSUMÉ RAPIDE

- ✓ Le transistor BJT permet d'amplifier ou de commuter un signal.
- ✓ Le courant de base (petit) contrôle le courant collecteur (grand).
- ✓ En coupeure : transistor OFF. En saturation : transistor ON.
- ✓ Toujours calculer la résistance de base pour assurer une saturation sûre.
- ✓ Les transistors sont au cœur de nombreux circuits électroniques et robotiques.



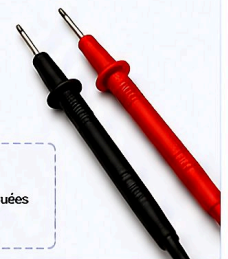
10 CONSEIL PRATIQUE

Utilisez toujours un multimètre pour vérifier les tensions ($V_{BE} \approx 0.7 \text{ V}$) et le mode de fonctionnement de votre transistor. Cela évite les erreurs et protège vos composants.



SÉCURITÉ

Ne dépassez jamais les limites de courant et de tension indiquées dans le datasheet du transistor.



2

CHAPITRE 2

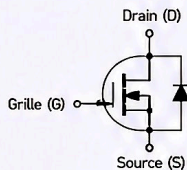
Électronique
fondamentale

2.29

MOSFET ET APPLICATIONS

Un transistor pour l'électronique moderne

Symbole MOSFET canal N

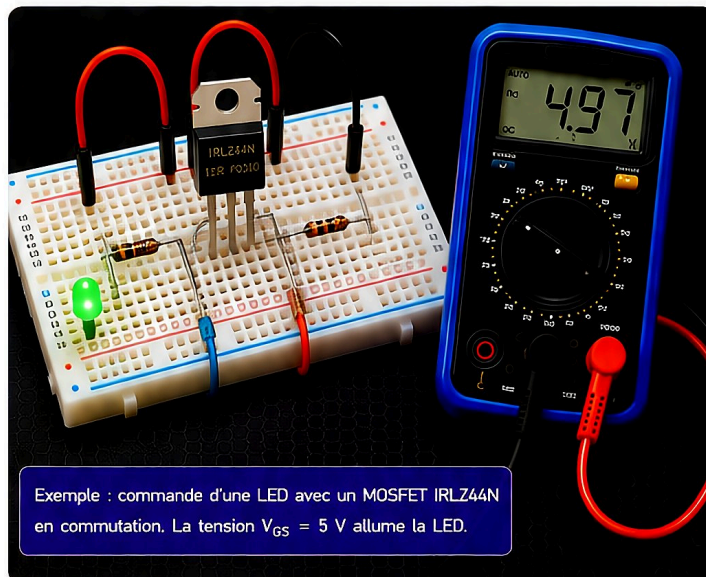


Le MOSFET (Metal-Oxide-Semiconductor Field-Effect Transistor) est un transistor à effet de champ à très haute impédance d'entrée et à faible résistance à l'état passant.

Le MOSFET est largement utilisé comme interrupteur électronique et dans les amplificateurs de puissance à haute efficacité.



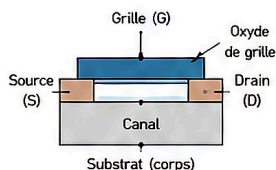
Sa commande en tension (grille) permet une très faible consommation et des commutations rapides.



Exemple : commande d'une LED avec un MOSFET IRLZ44N en commutation. La tension $V_{GS} = 5\text{ V}$ allume la LED.

1 QU'EST-CE QU'UN MOSFET ?

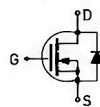
Le MOSFET est un transistor à effet de champ. Un champ électrique appliqué à la grille contrôle le courant entre le drain (D) et la source (S).



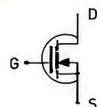
Courant contrôlé par la tension de grille, presque aucun courant d'entrée.

2 TYPES DE MOSFET

Canal N (le plus courant)



Canal P

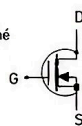


Canal N : s'active avec une tension positive sur la grille.

3 FONCTIONNEMENT (CANAL N)

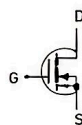
Coupeure (OFF)

$V_{GS} < V_{TH} \rightarrow$ Pas de canal formé
 $I_D \approx 0$ (transistor ouvert)



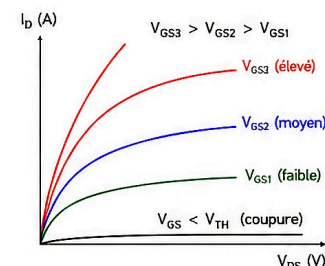
Conduction (ON)

$V_{GS} \geq V_{TH} \rightarrow$ Canal formé
Le courant circule entre D et S (transistor fermé)



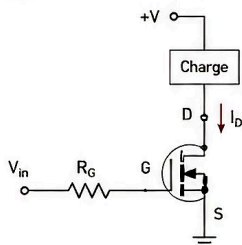
V_{TH} : tension de seuil (threshold) typique de 1 à 4 V selon le modèle.

4 CARACTÉRISTIQUE $I_D - V_{DS}$



- Région ohmique (linéaire) : faibles V_{DS}
- Région de saturation : V_{DS} élevé
- V_{GS} plus grand $\rightarrow I_D$ plus grand

5 COMMUTATION LOW-SIDE



Le MOSFET est placé entre la charge et la masse (0 V). Simple et très utilisé.

Idéal pour piloter des LED, moteurs, relais et charges DC.

6 RÉGIME LINÉAIRE ($R_{DS(on)}$)

Quand le MOSFET est ON, il se comporte comme une résistance.

$$V_{DS} = I_D \times R_{DS(on)}$$

Avantages

- ✓ Faible perte de puissance
- ✓ Faible échauffement
- ✓ Haute efficacité
- ✓ Idéal pour les applications de puissance

7 CALCUL DES PERTES

a) Pertes en conduction

$$P_{cond} = I_D^2 \times R_{DS(on)}$$

b) Pertes totales (approx.)

$$P_{tot} = P_{cand} + P_{switch}$$

Choisir un MOSFET avec $R_{DS(on)}$ la plus faible possible pour réduire les pertes.

Vérifier le courant max et la dissipation thermique (P_D max).

8 APPLICATIONS COURANTES

- Commandes de moteurs DC
- Convertisseurs DC-DC
- Alimentations à découpage (SMPS)
- Drivers de LED haute puissance
- Amplificateurs audio (classe D)
- Gestion de batteries
- Circuits de protection électronique



9 MOSFET POPULAIRES ET CARACTÉRISTIQUES

Modèle	Type	$V_{DS\text{ max}}$	$I_D\text{ max}$	$R_{DS(on)\text{ typ.}}$	Puissance	Boîtier
IRLZ44N	N-Channel	55 V	47 A	0,022 Ω	94 W	TO-220
IRF540N	N-Channel	100 V	33 A	0,077 Ω	125 W	TO-220
A03400	N-Channel	30 V	5,7 A	0,050 Ω	1,5 W	SOT-23
FQP30N06L	N-Channel	60 V	30 A	0,035 Ω	40 W	TO-220

Toujours consulter la datasheet avant utilisation et respecter les limites thermiques et électriques.

10 À RETENIR

- ✓ Le MOSFET est commandé en tension, pas en courant.
- ✓ $V_{GS} > V_{TH} \rightarrow$ le transistor conduit.
- ✓ Faible $R_{DS(on)} \rightarrow$ faible perte de puissance.
- ✓ Idéal pour les commutations rapides et les hautes puissances.
- ✓ Bien choisir le MOSFET selon la tension, le courant et la max dissipation.
- ✓ Utiliser un dissipateur thermique si nécessaire.



Conseil pratique

Utilisez une résistance de grille (R_G) entre 47 Ω et 220 Ω pour éviter les oscillations et limiter les pics de courant.



2

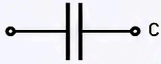
CHAPITRE 2 Électronique fondamentale

2.30

CONDENSATEURS ET APPLICATIONS

Stocker, filtrer et stabiliser l'énergie

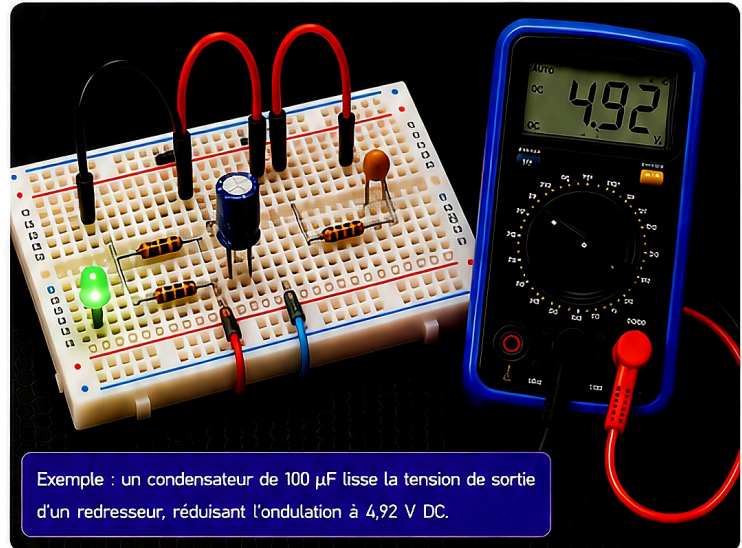
Symbole d'un condensateur



C : Capacité en Farads (F)
Couramment : μF (microfarads)
nF (nanofarads)

Le condensateur est un composant passif capable de stocker de l'énergie dans un champ électrique entre deux armatures séparées par un diélectrique.

Il est utilisé pour le filtrage, le découplage, le temporisation, le couplage de signaux et le lissage des alimentations.

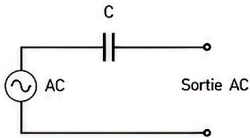


Exemple : un condensateur de 100 μF lisse la tension de sortie d'un redresseur, réduisant l'ondulation à 4,92 V DC.

1 RÔLE DU CONDENSATEUR

Un condensateur peut :

- Stocker de l'énergie
- Bloquer le courant continu (DC)
- Laisser passer le courant alternatif (AC)
- Réduire les variations de tension



Il se charge avec une tension et se décharge pour maintenir la stabilité du circuit.

2 TYPES DE CONDENSATEURS

Type	Symbole	Caractéristiques	Polarité
Céramique		Petite valeur, haute fréquence	Non
Film		Bonne stabilité, faible perte	Non
Électrolytique		Grande capacité, faible coût	Oui
Tantale		Compact, stable, haute capacité	Oui

Respecter la polarité des condensateurs électrolytiques et tantales.

3 CAPACITÉ ET ÉNERGIE STOCKÉE

La capacité C dépend de la géométrie et du diélectrique.

Formule de la capacité :

$$C = \frac{\epsilon \times A}{d}$$

ϵ : permittivité du diélectrique (F/m)
A : surface des armatures (m^2)
d : distance entre les armatures (m)

Énergie stockée :

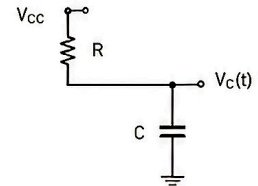
$$E = \frac{1}{2} CV^2$$

E en joules (J), V en volts (V).

L'énergie croît avec le carré de la tension.

4 CHARGE ET DÉCHARGE (RC)

Circuit RC en charge :



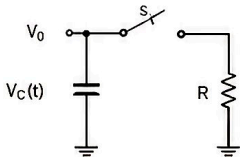
Tension aux bornes du condensateur :

$$V_C(t) = V_{CC} (1 - e^{-t/RC})$$

Constante de temps : $\tau = RC$

Après 5τ , le condensateur est pratiquement chargé.

5 CIRCUIT RC EN DÉCHARGE



Tension aux bornes :

$$V_C(t) = V_0 e^{-t/RC}$$

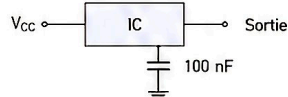
Le condensateur se décharge exponentiellement.

À $t = 5\tau$, la tension est $\approx 0,7\%$ de la valeur initiale.

6 DÉCOUPLAGE ET FILTRAGE

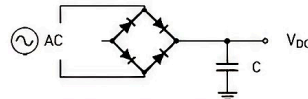
a) Découplage (bypass)

Un petit condensateur (100 nF) placé près d'un circuit intégré élimine les parasites haute fréquence.



b) Filtrage d'une alimentation

Redresseur + condensateur = tension lissée.



Plus la capacité est grande, plus l'ondulation résiduelle est faible.

7 CODES DE MARQUAGE

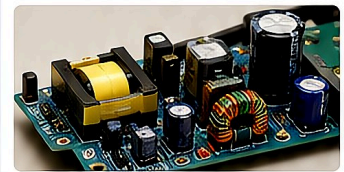
Exemple	Valeur
104	$10 \times 10^4 \text{ pF} = 100 \text{ nF}$
105	$10 \times 10^5 \text{ pF} = 1 \mu\text{F}$
10 μF 25V	10 μF , 25 volts
47 μF 50V	47 μF , 50 volts
2R2	2,2 μF
473	47 nF



Toujours vérifier la valeur et la tension maximale avant utilisation.

8 APPLICATIONS COURANTES

- Lissage des alimentations DC
- Temporisation (circuits RC)
- Couplage de signaux audio
- Filtrage passe-bas / passe-haut
- Oscillateurs (avec résistances et autres composants)
- Stabilisation de tension



Indispensable dans presque tous les circuits électroniques.

9 VALEURS DE CONDENSATEURS COURANTES

Unité	Symbole	Exemples	Équivalent
Picofarad	pF	10 pF, 220 pF	10^{-12} F
Nanofarad	nF	100 nF, 470 nF	10^{-9} F
Microfarad	μF	1 μF , 47 μF , 100 μF	10^{-6} F
Millifarad	mF	1 mF, 10 mF	10^{-3} F
Farad	F	1 F, 10 F	1 F

Utiliser des unités adaptées pour éviter les erreurs de câblage.

10 À RETENIR

- ✓ Le condensateur stocke de l'énergie sous forme de champ électrique.
- ✓ Il bloque le DC et laisse passer l'AC.
- ✓ La tension et le courant suivent une loi exponentielle dans les circuits RC.
- ✓ Respecter la polarité des condensateurs polarisés.
- ✓ Utilisé pour le filtrage, le découplage, la temporisation et la stabilisation.
- ✓ Choisir la bonne valeur et la bonne tension.

Conseil pratique

Utilisez un multimètre pour vérifier la tension et la polarité avant d'installer un condensateur électrolytique.



2

CHAPITRE 2

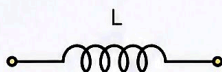
Électronique
fondamentale

2.31

BOBINES (INDUCTANCES) ET APPLICATIONS

Stocker l'énergie dans un champ magnétique

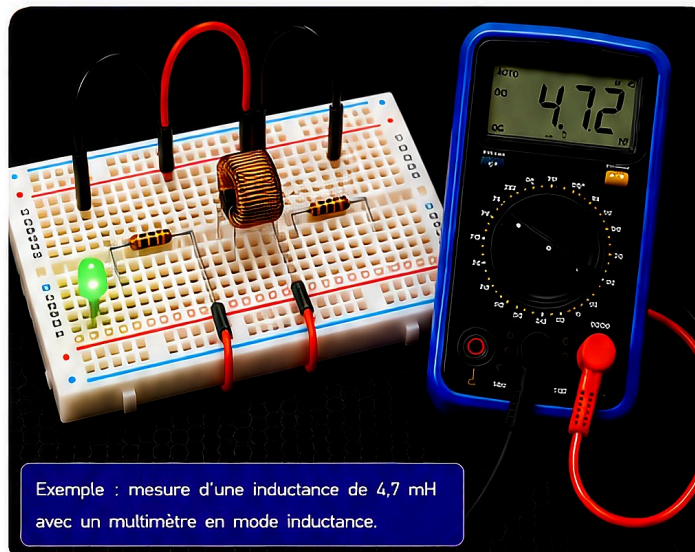
Symbole d'une bobine (inductance)



L : Inductance (Henrys, H)
mH : millihenry (10^{-3} H)
 μ H : microhenry (10^{-6} H)

Une bobine est un composant passif qui stocke de l'énergie sous forme de champ magnétique lorsqu'un courant la traverse.

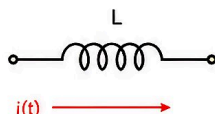
Elle s'oppose aux variations de courant et est utilisée dans de nombreuses applications : filtres, oscillateurs, convertisseurs et alimentations.



Exemple : mesure d'une inductance de 4,7 mH avec un multimètre en mode inductance.

1 RÔLE D'UNE BOBINE

- Stocker de l'énergie magnétique
- S'opposer aux variations de courant
- Laisser passer le courant continu (DC)
- Filtrer, lisser ou couper certaines fréquences (AC)



La bobine s'oppose aux changements rapides du courant.

2 UNITÉ ET SYMBOLES

Symbole	Unité	Abréviation
Henry		H
Millihenry		mH
Microhenry		μ H

Exemples :

- 1 mH = 10^{-3} H
- 1 μ H = 10^{-6} H
- 1 H = 1000 mH = 1 000 000 μ H

Plus l'inductance est grande, plus la bobine stocke d'énergie.

3 TENSION AUX BORNES

La tension aux bornes d'une bobine :

$$v_L(t) = L \frac{di(t)}{dt}$$

où :

- $v_L(t)$: tension (V)
- L : inductance (H)
- $\frac{di(t)}{dt}$: variation du courant (A/s)

La tension est proportionnelle à la vitesse de variation du courant.

4 ÉNERGIE STOCKÉE

L'énergie magnétique stockée :

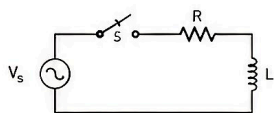
$$W = \frac{1}{2} L I^2$$

où :

- W : énergie (Joules)
- L : inductance (H)
- I : courant (Ampères)

L'énergie augmente avec le carré du courant.

5 RÉPONSE À UN ÉCHELON DE TENSION CIRCUIT RL SÉRIE



Constante de temps :

$$\tau = \frac{L}{R}$$

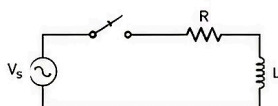
Réponse du courant :

$$i(t) = \frac{V_s}{R} (1 - e^{-t/\tau})$$

Le courant croît exponentiellement vers sa valeur finale.

6 RÉPONSE À L'OUVERTURE

Circuit RL lors de l'ouverture du switch :



Courant décroissant :

$$i(t) = I_0 e^{-t/\tau}$$

où I_0 est le courant juste avant l'ouverture.

La bobine maintient le courant le plus longtemps possible.

7 IMPÉDANCE D'UNE BOBINE (AC)

En régime sinusoïdal (fréquence f) :

$$X_L = \omega L = 2\pi f L$$

où :

- X_L : réactance inductive (Ω)
- ω : pulsation (rad/s)
- f : fréquence (Hz)

L'impédance complexe :

$$Z_L = j\omega L$$

Une bobine laisse passer les hautes fréquences et bloque les basses.

8 APPLICATIONS COURANTES

- Filtres passe-bas / passe-haut
- Convertisseurs DC-DC (buck, boost)
- Alimentations à découpage (SMPS)
- Oscillateurs et circuits résonants
- Lissage du courant dans les drivers de moteurs
- Protection contre les surtensions (en association avec diodes)



Indispensable dans les alimentations et les circuits de puissance.

9 VALEURS TYPIQUES D'INDUCTANCES

Composant	Symbole	Valeurs typiques	Équivalent
Inductance CMS (puce)		1 μ H – 100 μ H	10^{-6} – 10^{-4} H
Bobine axiale 1/4 W		10 μ H – 10 mH	10^{-5} – 10^{-2} H
Bobine de puissance (radiale)		100 μ H – 10 mH	10^{-4} – 10^{-2} H
Inductance de filtrage		1 mH – 100 mH	10^{-3} – 10^{-1} H
Self de mode commun		100 μ H – 10 mH	10^{-4} – 10^{-2} H

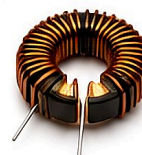
Toujours vérifier la valeur et le courant nominal avant utilisation.

10 À RETENIR

- ✓ La bobine stocke l'énergie dans un champ magnétique.
- ✓ La tension aux bornes dépend de la variation du courant : $v_L = L \frac{di}{dt}$.
- ✓ Elle s'oppose aux changements rapides du courant.
- ✓ En AC, son impédance augmente avec la fréquence : $X_L = 2\pi f L$.
- ✓ Utilisée dans les filtres, alimentations, oscillateurs et convertisseurs.
- ✓ Bien choisir l'inductance, le courant max et la résistance série (DCR).

Conseil pratique

Lors du test d'une bobine, utiliser un multimètre en mode inductance ou mesurer la tension aux bornes pour vérifier qu'elle fonctionne correctement.



2

CHAPITRE 2

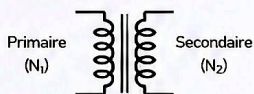
Électronique
fondamentale

2.32

TRANSFORMATEURS ET APPLICATIONS

Transférer l'énergie en courant alternatif

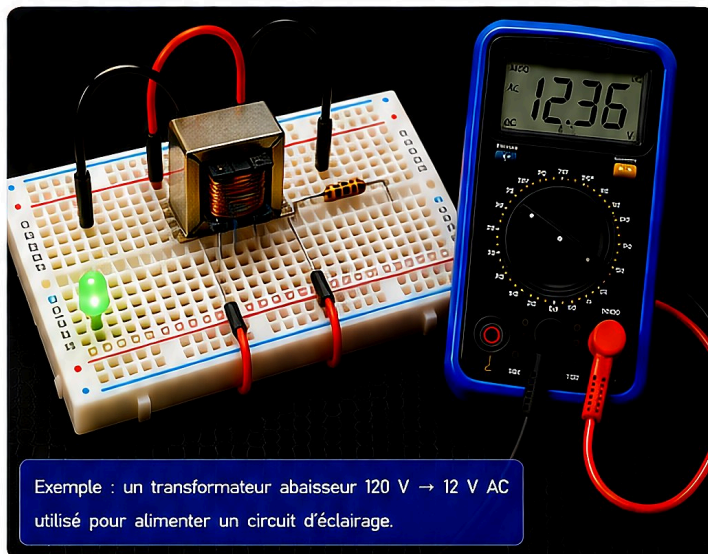
Symbole d'un transformateur



N_1 : nombre de spires du primaire
 N_2 : nombre de spires du secondaire

Un transformateur est un composant électromagnétique qui transfère l'énergie électrique d'un circuit AC à un autre par induction magnétique.

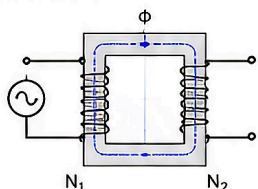
Il permet d'augmenter (élévateur) ou de diminuer (abaisseur) la tension AC tout en conservant la fréquence.



Exemple : un transformateur abaisseur 120 V → 12 V AC utilisé pour alimenter un circuit d'éclairage.

1 PRINCIPE DE FONCTIONNEMENT

Le courant alternatif dans le primaire (N_1) crée un flux magnétique variable Φ dans le noyau. Ce flux induit une tension dans le secondaire (N_2).



L'énergie est transférée sans connexion électrique directe entre les circuits.

2 RAPPORT DE TRANSFORMATION

La relation entre les tensions est :

$$\frac{V_1}{V_2} = \frac{N_1}{N_2}$$

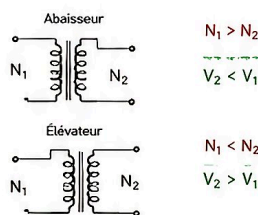
où :

- V_1 : tension efficace au primaire (V)
- V_2 : tension efficace au secondaire (V)
- N_1 : nombre de spires du primaire
- N_2 : nombre de spires du secondaire

Le rapport des tensions est égal au rapport des spires.

3 TYPES DE TRANSFORMATEURS

- Abaisseur** : $V_2 < V_1$ ($N_2 < N_1$)
Diminue la tension.
- Élévateur** : $V_2 > V_1$ ($N_2 > N_1$)
Augmente la tension.



Le courant et la tension varient en sens inverse.

4 COURANT ET PUISSANCE IDÉALS

En négligeant les pertes :

$$V_1 I_1 = V_2 I_2$$

où :

- I_1 : courant efficace au primaire (A)
- I_2 : courant efficace au secondaire (A)

$$\text{Donc : } \frac{I_1}{I_2} = \frac{N_2}{N_1}$$

Quand la tension augmente, le courant diminue, et inversement.

5 TRANSFORMATEUR IDÉAL

Hypothèses :

- Aucune perte de cuivre
- Aucune perte dans le noyau
- Couplage magnétique parfait

Dans ce cas :

$$\eta = \frac{P_{out}}{P_{in}} = 100\%$$

En réalité, les pertes réduisent légèrement le rendement.

6 RENDEMENT (η)

Le rendement d'un transformateur réel :

$$\eta = \frac{P_{out}}{P_{in}} \times 100\%$$

Où :

- P_{in} : puissance d'entrée (W)
- P_{out} : puissance de sortie (W)

Les rendements typiques :

Petits transformateurs : 85 % – 95 %
Transformateurs de puissance : 95 % – 99 %

Moins il y a de pertes, meilleur est le rendement.

7 EXEMPLE NUMÉRIQUE

Transformateur abaisseur :

$N_1 = 1200$ spires, $N_2 = 120$ spires
 $V_1 = 120$ V AC

a) Tension secondaire :

$$V_2 = V_1 \times \frac{N_2}{N_1} = 120 \times \frac{120}{1200} = 12 \text{ V}$$

b) Si le courant secondaire est 2 A :

$$I_1 = I_2 \times \frac{N_2}{N_1} = 2 \times \frac{120}{1200} = 0,2 \text{ A}$$

La tension est réduite à 12 V, et le courant au primaire est 0,2 A.

8 APPLICATIONS COURANTES

- Alimentations AC/DC (chargeurs)
- Isolation électrique
- Adaptation d'impédance audio
- Distribution et transport d'énergie
- Convertisseurs et onduleurs
- Circuits de mesure et de protection



Les transformateurs sont indispensables dans la vie quotidienne.

9 CARACTÉRISTIQUES ET VALEURS TYPIQUES

Type	Puissance typique	Fréquence typique	Tension primaire (V)	Tension secondaire (V)	Utilisation
Petit transformateur de signal	0,1 – 5 VA	50 – 60 Hz	6 – 24	6 – 24	Circuits audio, électronique
Transformateur d'alimentation	5 – 50 VA	50 – 60 Hz	120 / 230	6 – 24	Alimentations électroniques
Transformateur de puissance	100 VA – 10 kVA	50 – 60 Hz	120 / 230 / 480	12 – 240	Distribution électrique
Transformateur d'isolement	10 – 1000 VA	50 – 60 Hz	120 / 230	120 / 230	Sécurité, protection

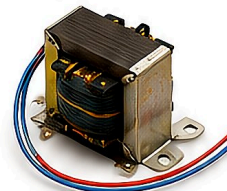
Toujours vérifier la puissance nominale et la tension avant utilisation.

10 À RETENIR

- Le transformateur transfère l'énergie par induction magnétique.
- Le rapport des tensions est égal au rapport du nombre de spires.
- La puissance est presque conservée ($V_1 I_1 \approx V_2 I_2$).
- Les transformateurs abaisseurs réduisent la tension, les élévateurs l'augmentent.
- Le rendement dépend des pertes dans le cuivre et le noyau.
- Utiliser le bon transformateur assure la sécurité et l'efficacité du circuit.

Conseil pratique

Choisissez un transformateur dont la tension et la puissance conviennent à votre application, avec une marge de sécurité de 20 à 30 %.



2

CHAPITRE 2

Électronique
fondamentale

2.33

DIODES ET REDRESSEMENT ET APPLICATIONS

Contrôler le courant en un seul sens

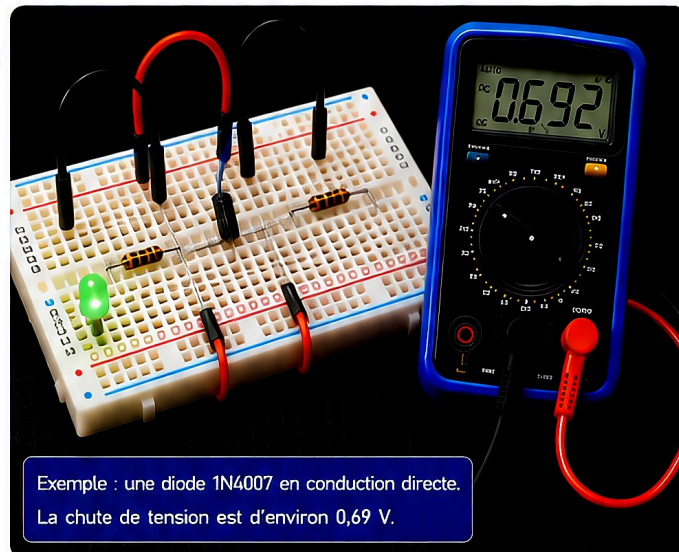
Symbole d'une diode

Anode (A) Cathode (K)

La diode laisse passer le courant de l'anode (A) vers la cathode (K) et le bloque dans le sens inverse.

La diode est un composant électronique à deux bornes qui conduit le courant dans un seul sens et le bloque dans l'autre.

Elle est largement utilisée pour le redressement, la protection et la régulation de circuits.



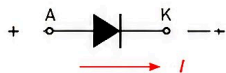
Exemple : une diode 1N4007 en conduction directe.

La chute de tension est d'environ 0,69 V.

1 FONCTIONNEMENT D'UNE DIODE

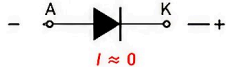
Polarisation directe

Anode (+) plus positive que cathode (-) : la diode conduit.



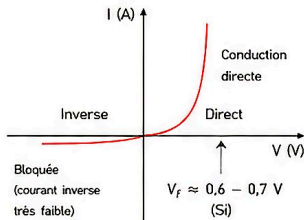
Polarisation inverse

Anode (-) plus négative que cathode (+) : la diode bloque.



La diode est comme une valve électronique : un seul sens autorisé.

2 CARACTÉRISTIQUE I-V D'UNE DIODE



- En direct : la diode conduit après V_F .
- En inverse : courant de fuite très faible jusqu'à la tension de claquage.

Pour une diode au silicium, la chute de tension directe $\approx 0,6 - 0,7$ V.

3 SPÉCIFICATIONS IMPORTANTES

- V_F : tension directe ($\approx 0,6 - 0,7$ V pour Si)
- I_{Fmax} : courant direct maximum
- V_{RRM} : tension inverse répétitive max
- I_R : courant de fuite inverse

Paramètre	Symbole	Unité
Tension directe	V_F	V
Courant direct max	I_{Fmax}	A
Tension inverse max	V_{RRM}	V
Courant inverse	I_R	μA

Toujours vérifier la datasheet avant d'utiliser une diode.

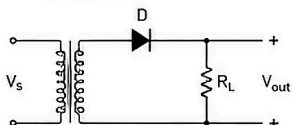
4 TYPES DE DIODES

- Diode de redressement (ex : 1N4007)
- Diode Zener
- Diode Schottky
- LED (diode électroluminescente)
- Diode de signal (ex : 1N4148)
- Diode PIN et photodiode
- Diode varicap

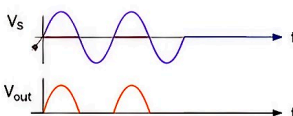


Chaque type de diode a une application spécifique.

5 REDRESSEMENT SIMPLE DEMI-ONDE

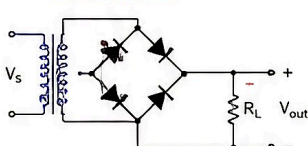


Forme d'onde :

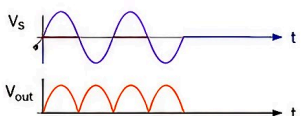


Une seule alternance est utilisée (rendement faible).

6 REDRESSEMENT DOUBLE ALTERNANCE PONT DE GRAETZ



Forme d'onde :



Les deux alternances sont utilisées. Tension plus élevée et plus efficace.

7 TENSION MOYENNE DE SORTIE

Pour un redressement double alternance :

$$V_{DC} \approx \frac{2V_m}{\pi} - 2V_F$$

où :

V_m : amplitude de la tension AC (V)

V_F : chute de tension d'une diode (V)

Exemple :

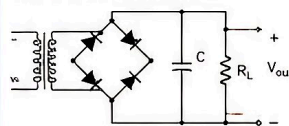
Si $V_m = 10$ V et $V_F = 0,7$ V

$$V_{DC} \approx \frac{2 \times 10}{\pi} - 2 \times 0,7 \approx 5,87$$

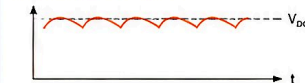
La tension réelle sera légèrement plus faible à cause des pertes.

8 CIRCUIT DE FILTRAGE

Un condensateur en parallèle avec la charge lisse la tension redressée.



Forme d'onde de sortie :



Plus la capacité C est grande, plus l'ondulation est faible.

9 DIODES DE REDRESSEMENT COURANTES

	Modèle	I_{FAV} (A)	V_{RRM} (V)	I_{FSM} (A)	V_F (V)	Boîtier
	1N4001	1	50	30	1,0	DO-41
	1N4004	1	400	30	1,0	DO-41
	1N4007	1	1000	30	1,0	DO-41
	1N5819 (Schottky)	1	40	25	0,3 - 0,4	DO-41
	1N4148 (signal)	0,2	100	1	1,0	DO-35
	LED rouge	0,02	2	0,1	1,8 - 2,2	5 mm

Utiliser une diode avec des spécifications supérieures à vos besoins.

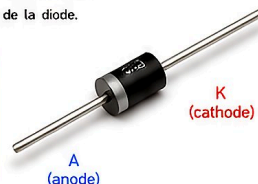
10 À RETENIR

- La diode conduit dans un seul sens et bloque dans l'autre.
- La chute de tension directe pour une diode au silicium est $\approx 0,6 - 0,7$ V.
- Les diodes sont essentielles pour redresser le courant AC en DC.
- Le pont de Graetz utilise les deux alternances et fournit une meilleure tension.
- Un condensateur de filtrage réduit l'ondulation de la tension de sortie.
- Toujours respecter les limites de courant et de tension de la diode.



Conseil pratique

Vérifiez la polarité avant d'installer une diode. Une inversion peut endommager le composant ou empêcher le circuit de fonctionner.



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

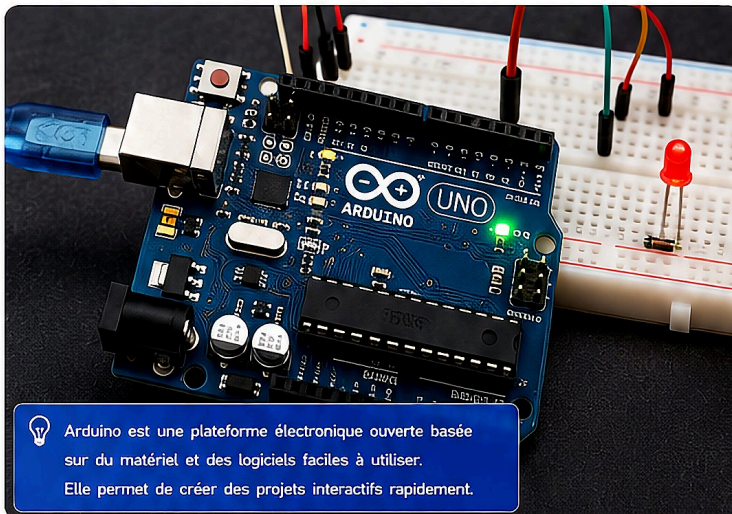
3.1

INTRODUCTION À ARDUINO

La plateforme open-source pour la robotique et l'électronique

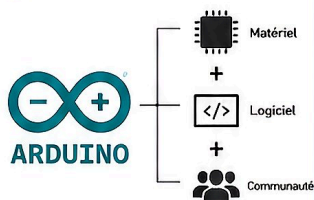
Arduino est une carte électronique programmable qui permet de contrôler des capteurs, des actionneurs (moteurs, LEDs, servos, etc.) et d'interagir avec le monde réel.

Elle est largement utilisée dans l'éducation, la robotique, l'automatisation, l'IoT et les projets créatifs.



Arduino est une plateforme électronique ouverte basée sur du matériel et des logiciels faciles à utiliser. Elle permet de créer des projets interactifs rapidement.

1 QU'EST-CE QU'ARDUINO ?



- **Matériel** : cartes électroniques
- **Logiciel** : IDE Arduino (gratuit)
- **Communauté** : partage de projets, bibliothèques et exemples
- **Open-source** : libre et extensible

Arduino est conçu pour être simple à utiliser, même si vous débutez en électronique et en programmation.

2 CARACTÉRISTIQUES PRINCIPALES

- **Facile à programmer**
Langage simple basé sur C/C++ avec une syntaxe simplifiée.
- **Open-source**
Matériel et logiciel ouverts, modifiables et extensibles.
- **Grande communauté**
Des milliers de projets, tutoriels et bibliothèques disponibles.
- **Polyvalent**
Convient à l'électronique, la robotique, l'IoT, l'impression 3D, et bien plus encore.

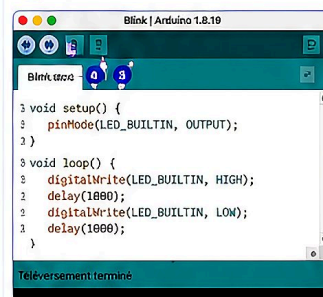
Avec Arduino, vous pouvez passer de l'idée au prototype en quelques minutes seulement !

3 À QUOI SERT ARDUINO ?

- Lire des capteurs (température, distance, lumière, etc.)
- Commander des actionneurs (LEDs, moteurs, relais, servomoteurs, etc.)
- Créer des systèmes interactifs et autonomes
- Communiquer avec d'autres appareils (Bluetooth, Wi-Fi, IoT, etc.)
- Prototyper et automatiser des projets rapidement

Arduino transforme vos idées en projets concrets et fonctionnels.

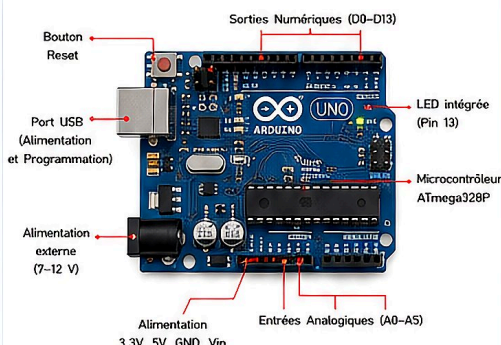
4 L'ENVIRONNEMENT ARDUINO IDE



- 1 Vérifier le code
- 2 Téléverser le programme
- 3 Nouveau / Ouvrir / Enregistrer
- 4 Moniteur série
- 5 Zone d'édition du code

L'IDE Arduino est disponible gratuitement pour Windows, macOS et Linux.

5 CARTE ARDUINO UNO – APERÇU



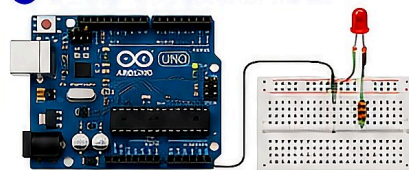
La carte Arduino UNO est idéale pour apprendre et réaliser la plupart des projets électroniques et robotiques.

6 SPÉCIFICATIONS DE L'ARDUINO UNO

Microcontrôleur	ATmega328P
Tension de fonctionnement	5 V
Tension d'entrée (recommandée)	7 – 12 V
Tension d'entrée (limites)	6 – 20 V
Broches numériques	14 (dont 6 PWM)
Entrées analogiques	6
Courant par broche I/O	20 mA
Mémoire Flash	32 KB (2 KB pour bootloader)
SRAM	2 KB
EEPROM	1 KB
Fréquence d'horloge	16 MHz

Vous pouvez alimenter l'UNO via USB ou via une source externe (batterie, alimentation, etc.).

7 EXEMPLE : FAIRE CLIGNOTER UNE LED



1. LED (anode) → D13 via une résistance 220 Ω
2. LED (cathode) → GND

Code Arduino :

```
void setup() {
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH); // Allume la LED
  delay(1000);            // Attend 1 seconde
  digitalWrite(13, LOW);  // Éteint la LED
  delay(1000);            // Attend 1 seconde
}
```

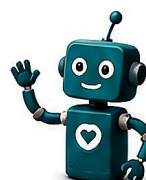
C'est le premier programme que tout débutant Arduino réalise !

8 POUR COMMENCER

1. Télécharger l'IDE Arduino (arduino.cc)
2. Connecter la carte Arduino via USB
3. Écrire ou ouvrir un exemple
4. Téléverser le code dans la carte
5. Observer le résultat et expérimenter !

9 À RETENIR

- ✓ Arduino est une plateforme open-source simple et puissante.
- ✓ Elle permet d'interagir avec le monde réel facilement.
- ✓ La carte Arduino UNO est idéale pour débuter.
- ✓ Apprendre par la pratique est la meilleure façon de progresser.
- ✓ La communauté Arduino est là pour vous aider !



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

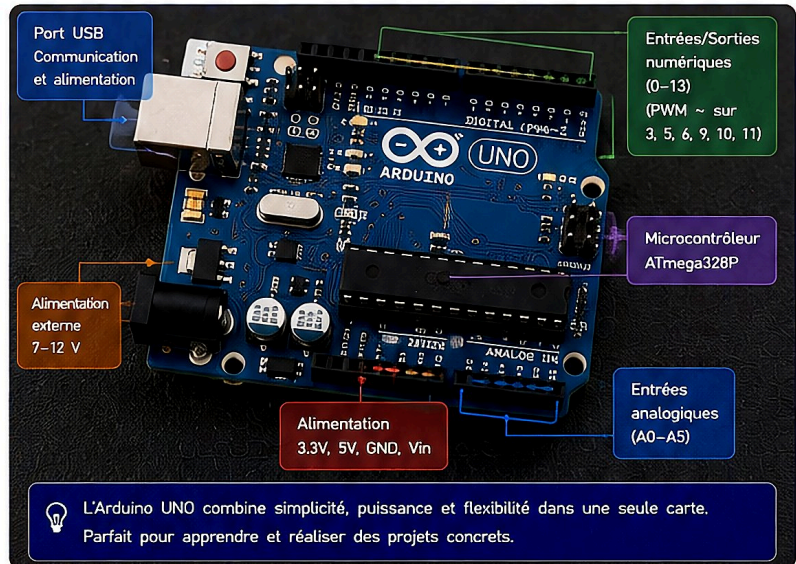
3.2

CARTE À ARDUINO UNO

Découvrir la carte la plus populaire pour l'électronique et la robotique

L'Arduino UNO est une carte à microcontrôleur basée sur l'ATmega328P. Elle possède tout ce qu'il faut pour commencer à créer, tester et contrôler des projets électroniques et robotiques.

Simple, fiable et compatible avec des milliers de ressources, elle est idéale pour les débutants.

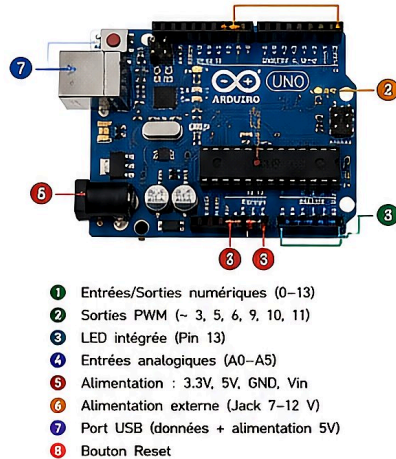


1 SPÉCIFICATIONS CLÉS

Microcontrôleur	ATmega328P
Tension de fonctionnement	5 V
Tension d'entrée (recommandée)	7 – 12 V
Tension d'entrée (limites)	6 – 20 V
Entrées/Sorties numériques	14 (dont 6 PWM)
Entrées analogiques	6
Courant par broche I/O	20 mA
Courant 3.3V	50 mA
Mémoire Flash	32 KB (0.5 KB pour bootloader)
SRAM	2 KB
EEPROM	1 KB
Fréquence d'horloge	16 MHz

Ces caractéristiques permettent à l'UNO de piloter capteurs, moteurs, LED, écrans et bien plus encore.

2 BROCHAGE ET FONCTIONS



Le schéma ci-dessus montre l'emplacement des broches et leurs fonctions principales.

3 ALIMENTATION DE LA CARTE

L'Arduino UNO peut être alimenté de plusieurs façons :

Via USB
5 V fournis par le port USB de l'ordinateur.

Via jack d'alimentation
7 à 12 V DC (recommandé). Centre positif (+).

Via la broche Vin
7 à 12 V DC sur la broche Vin de la carte.

⚠ Ne jamais appliquer plus de 20 V sur la broche Vin ou le jack d'alimentation.

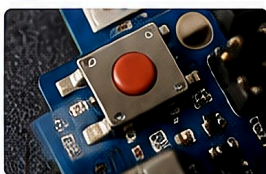
4 LED INTÉGRÉE (PIN 13)



La LED marquée "L" est connectée à la broche numérique 13. Elle est très utile pour les tests et le débogage rapide.

Si la LED clignote, c'est que votre programme fonctionne correctement !

5 BOUTON RESET



Le bouton RESET redémarre le microcontrôleur et relance votre programme depuis le début. Très pratique pour tester rapidement vos codes.

Vous pouvez aussi utiliser la fonction `reset()` dans votre programme.

6 COMMUNICATION



Ces protocoles permettent de connecter des capteurs, écrans, modules, etc.

7 EXEMPLE : FAIRE CLIGNOTER LA LED

```
// Programme : clignoter la LED intégrée
void setup() {
  pinMode(13, OUTPUT); // Pin 13 en sortie
}
void loop() {
  digitalWrite(13, HIGH); // Allume la LED
  delay(500); // Attend 0.5 s
  digitalWrite(13, LOW); // Éteint la LED
  delay(500); // Attend 0.5 s
}
```

- Téléversez le code dans la carte.
- La LED "L" clignotera toutes les 0.5 secondes.

C'est souvent le premier programme de tout débutant Arduino !

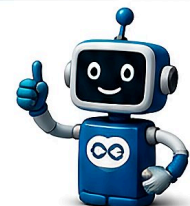
8 POURQUOI CHOISIR L'ARDUINO UNO ?



L'Arduino UNO est le point de départ parfait pour entrer dans le monde de la robotique, de l'électronique et de l'innovation !

9 À RETENIR

- ✓ L'Arduino UNO est basée sur le microcontrôleur ATmega328P.
- ✓ Elle dispose de 14 entrées/sorties numériques (dont 6 PWM) et 6 entrées analogiques.
- ✓ Elle peut être alimentée via USB ou une source externe 7-12 V.
- ✓ Elle est simple à utiliser, idéale pour les débutants.
- ✓ Elle est compatible avec une énorme communauté et de nombreux projets.



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

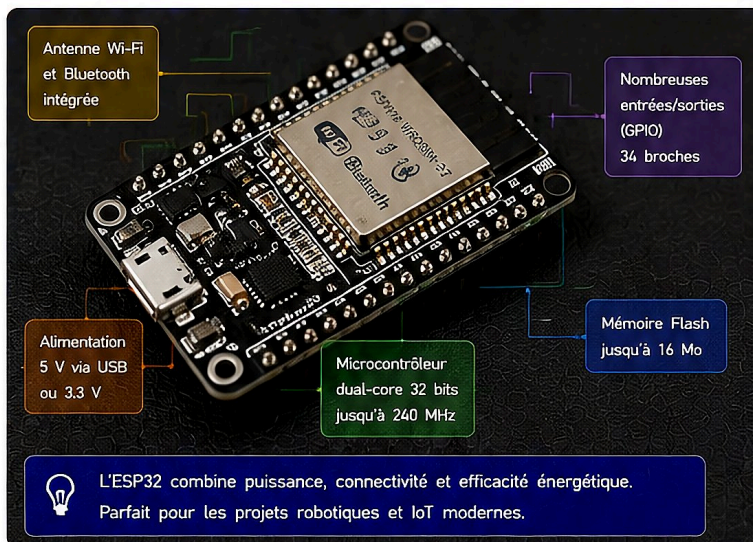
3.3

ESP32 ET MICROCONTRÔLEURS MODERNES

Plus de puissance, plus de connectivité pour des projets avancés

L'ESP32 est un microcontrôleur moderne qui intègre le Wi-Fi et le Bluetooth. Il offre plus de mémoire, plus de broches et une vitesse de traitement supérieure à l'Arduino UNO.

Il est idéal pour les projets IoT, la robotique connectée, le traitement de données et les applications temps réel.



L'ESP32 combine puissance, connectivité et efficacité énergétique. Parfait pour les projets robotiques et IoT modernes.

1 COMPARAISON : UNO vs ESP32

Caractéristique	Arduino UNO	ESP32
Microcontrôleur	ATmega328P	Xtensa LX6 dual-core
Fréquence	16 MHz	Jusqu'à 240 MHz
Tension logique	5 V	3.3 V
Broches E/S	14 (dont 6 PWM)	34 GPIO (dont 16 PWM)
Entrées analogiques	6	18 (12 bits)
Mémoire Flash	32 KB	Jusqu'à 16 MB
SRAM	2 KB	520 KB
Connectivité	—	Wi-Fi, Bluetooth (BLE)

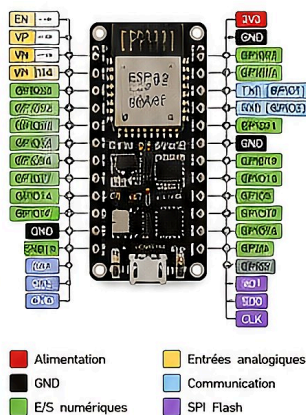
ESP32 offre beaucoup plus de ressources pour des projets avancés et connectés.

2 CARACTÉRISTIQUES PRINCIPALES

- Wi-Fi intégré**
Connexion réseau sans fil.
- Bluetooth BLE intégré**
Communication sans fil à faible consommation.
- Double cœur 32 bits**
Puissance de calcul élevée pour le multitâche.
- Grande mémoire**
Jusqu'à 16 Mo de Flash et 520 Ko de RAM.
- Beaucoup d'E/S**
34 broches GPIO, PWM, I2C, SPI, UART, ADC, DAC, etc.

ESP32 est un microcontrôleur très complet et polyvalent.

3 BROCHAGE DE L'ESP32 DEVKIT



Chaque broche peut avoir plusieurs fonctions. Vérifiez toujours la datasheet.

4 ALIMENTATION DE L'ESP32

Via USB (recommandé)

- Connectez le câble USB à l'ordinateur.
- Le régulateur fournit 5 V puis 3.3 V à la carte.

Via broche VIN

- 5 à 12 V DC sur la broche VIN.
- Le régulateur interne fournit 3.3 V.

Attention !
Toutes les E/S sont en 3.3 V. Ne jamais appliquer 5 V directement sur les GPIO.

Alimentez correctement votre ESP32 pour éviter d'endommager la carte.

5 ENVIRONNEMENT DE PROGRAMMATION



L'ESP32 se programme avec l'Arduino IDE. Il suffit d'ajouter le support ESP32 dans le gestionnaire de cartes.

Étapes :

- Ouvrir Arduino IDE
- Aller dans Fichier > Préférences
- Ajouter l'URL du gestionnaire ESP32
- Installer la carte "ESP32 by Espressif Systems"

Interface simple et familière pour débuter rapidement.

6 PROGRAMME EXEMPLE : CLIGNOTER UNE LED



```
// Clignoter la LED intégrée (GPIO2)
void setup() {
  pinMode(2, OUTPUT);
}

void loop() {
  digitalWrite(2, HIGH);
  delay(500);
  digitalWrite(2, LOW);
  delay(500);
}
```

La LED intégrée est connectée à la broche GPIO2. Elle clignote toutes les 0.5 secondes.

Premier pas pour contrôler votre ESP32 !

7 APPLICATIONS COURANTES

- Projets IoT (Internet des Objets)
- Robotique connectée
- Domotique et maison intelligente
- Capteurs et télémétrie
- Affichage et interfaces web
- Traitement de données en temps réel
- Communication sans fil (Wi-Fi/BLE)

Les possibilités sont énormes avec l'ESP32 !

8 ESP32 VS AUTRES CARTES

	Arduino UNO	Arduino MEGA	ESP32 DEVKIT
Conn. sans fil	Non	Non	Wi-Fi + Bluetooth
Fréquence	16 MHz	16 MHz	Jusqu'à 240 MHz
Broches GPIO	14	54	34
Entrées analogiques	6	16	18 (12 bits)
Flash	32 KB	256 KB	Jusqu'à 16 MB
Idéal pour	Débutants simples	Projets avancés	IoT, Robotique connectée

ESP32 est le meilleur choix pour des projets modernes et connectés.

9 POURQUOI CHOISIR L'ESP32 ?



Haute performance



Connectivité intégrée



Grande mémoire



Faible consommation



Coût abordable



Open-source et communauté active

ESP32 est aujourd'hui l'un des microcontrôleurs les plus populaires au monde.

10 À RETENIR

- ESP32 est un microcontrôleur moderne et puissant.
- Il intègre le Wi-Fi et le Bluetooth pour des projets connectés.
- Il possède plus de mémoire, plus de broches et une meilleure performance qu'un Arduino UNO.
- Il se programme facilement avec l'Arduino IDE.
- Il est parfait pour les projets IoT, la robotique et les applications avancées.



3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

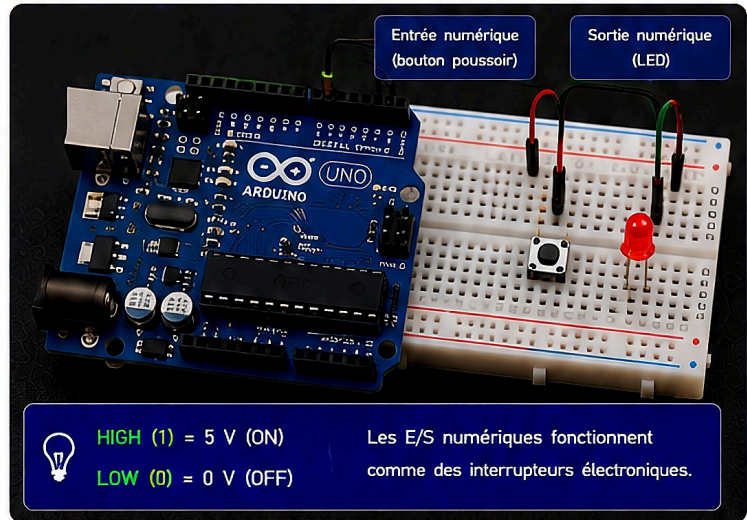
3.4

ENTRÉES ET SORTIES NUMÉRIQUES

Contrôler et lire des signaux ON ou OFF avec Arduino et ESP32

Les entrées et sorties numériques permettent à la carte de communiquer avec le monde extérieur à l'aide de signaux logiques : HIGH (1) ou LOW (0).

Elles sont essentielles pour lire des boutons, détecter des capteurs tout ou rien et commander des LEDs, relais, buzzer, moteurs, etc.



1 QU'EST-CE QU'UNE E/S NUMÉRIQUE ?

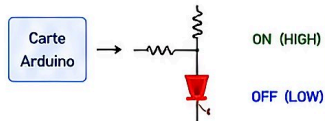
Entrée numérique :

Lit un signal 0 V (LOW) ou 5 V (HIGH).



Sortie numérique :

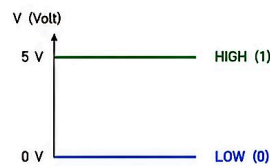
Envoie un signal 0 V (LOW) ou 5 V (HIGH).



C'est la base de nombreuses applications en robotique et en automatisation.

2 NIVEAUX LOGIQUES

État	Tension	Interprétation
HIGH (1)	≈ 5 V	ON, Vrai, Actif
LOW (0)	≈ 0 V	OFF, Faux, Inactif



La plupart des cartes Arduino et ESP32 fonctionnent en logique 5 V (3,3 V sur ESP32 en interne mais BROCHES tolèrent 3,3-5 V).

3 BROCHES NUMÉRIQUES

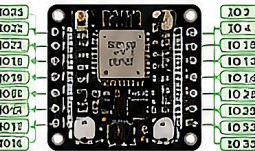
Arduino UNO

14 entrées/sorties numériques (0-13).



ESP32 DevKit

Nombreuses broches GPIO (ex : 0 à 23, 25 à 27, 32 à 33, etc.).



Choisissez n'importe quelle broche GPIO pour vos E/S numériques (sauf les broches réservées au démarrage).

4 EXEMPLES D'UTILISATION

Entrées numériques :

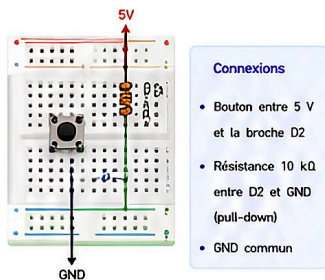
- Boutons poussoirs
- Interrupteurs
- Capteurs tout ou rien (fin de course, détecteurs)

Sorties numériques :

- LED
- Relais
- Buzzer
- Moteurs via driver

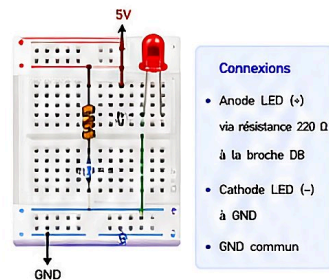
Permet de contrôler et d'interagir avec de nombreux composants.

5 SCHÉMA : LIRE UN BOUTON (ENTRÉE)



Quand le bouton est appuyé, la broche D2 lit HIGH (1). Sinon, elle lit LOW (0) grâce à la résistance.

6 SCHÉMA : COMMANDER UNE LED (SORTIE)



Quand D8 est HIGH (1), la LED s'allume. Quand D8 est LOW (0), elle s'éteint.

7 FONCTIONS ARDUINO DE BASE

Entrées :

- `pinMode(pin, INPUT);`
- `digitalRead(pin);`

Sorties :

- `pinMode(pin, OUTPUT);`
- `digitalWrite(pin, HIGH);`
- `digitalWrite(pin, LOW);`

Ces fonctions permettent de configurer et de lire/écrire sur les broches numériques.

8 EXEMPLE DE CODE

```
// LED sur la broche D8
const int ledPin = 8;
const int buttonPin = 2;
int buttonState = 0;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
}

void loop() {
  buttonState = digitalRead(buttonPin);
  if (buttonState == HIGH) {
    digitalWrite(ledPin, HIGH); // Allume LED
  } else {
    digitalWrite(ledPin, LOW); // Éteint LED
  }
}
```

Appuyez sur le bouton pour allumer la LED. Relâchez pour l'éteindre.

9 BONNES PRATIQUES

- Toujours relier les GND entre la carte et les composants.
- Utiliser une résistance de tirage (pull-up ou pull-down) pour éviter les lectures instables.
- Vérifier la tension maximale tolérée par les broches (5 V sur UNO, 3,3-5 V sur ESP32).
- Ne jamais appliquer plus de 5 V sur une broche d'ESP32, risque d'endommager la carte.
- Tester toujours vos entrées/sorties avec le Moniteur Série pour vérifier les états.

10 À RETENIR

- Les E/S numériques sont basées sur deux états : HIGH (1) ou LOW (0).
- Les entrées lisent des signaux, les sorties envoient des signaux.
- Arduino UNO possède 14 E/S numériques (0-13).
- ESP32 offre de nombreuses broches GPIO pour E/S numériques.
- Les fonctions `pinMode()`, `digitalRead()` et `digitalWrite()` sont essentielles.
- Elles permettent de créer des projets interactifs simples et puissants.



3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

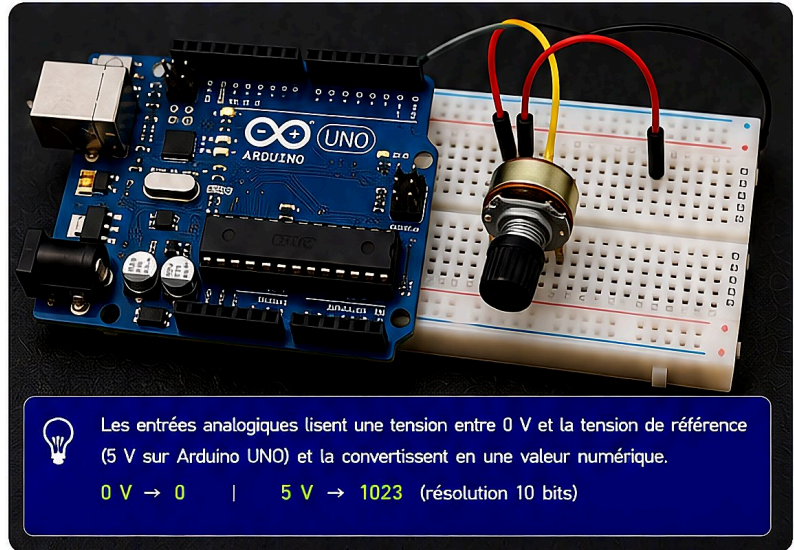
3.5

ENTRÉES ANALOGIQUES

Lire des valeurs variables avec Arduino et ESP32

Les entrées analogiques permettent de mesurer des tensions variables provenant de capteurs (température, lumière, position, etc.).

Elles convertissent la tension en une valeur numérique grâce au convertisseur analogique-numérique (CAN/ADC) intégré à la carte.

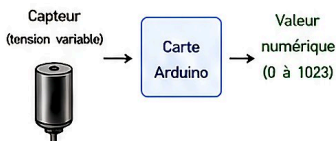


Les entrées analogiques lisent une tension entre 0 V et la tension de référence (5 V sur Arduino UNO) et la convertissent en une valeur numérique.

0 V → 0 | 5 V → 1023 (résolution 10 bits)

1 QU'EST-CE QU'UNE ENTRÉE ANALOGIQUE ?

Une entrée analogique lit une tension continue variable et la convertit en nombre numérique.



Idéal pour les capteurs qui ne donnent pas seulement ON ou OFF, mais une valeur changeante.

2 PLAGES DE TENSION ET RÉOLUTION

Par défaut, Arduino UNO utilise une référence de 5 V avec une résolution de 10 bits.

Tension d'entrée	Valeur numérique (approx.)
0 V	0
2,5 V	512
5 V	1023

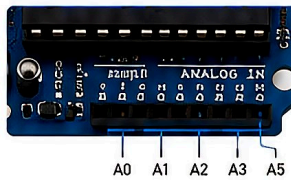
Formule de conversion :

$$\text{Valeur} = \frac{\text{Tension}_{\text{entrée}}}{\text{Tension}_{\text{référence}}} \times 1023$$

La plupart des cartes Arduino ont une résolution de 10 bits (0 à 1023).

3 BROCHES ANALOGIQUES

Arduino UNO possède 6 entrées analogiques : A0, A1, A2, A3, A4, A5.

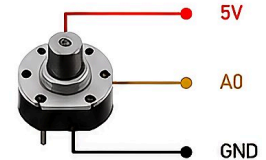


ESP32 possède jusqu'à 18 entrées analogiques (ADC1 et ADC2), selon le modèle.

Chaque broche peut lire une tension indépendante.

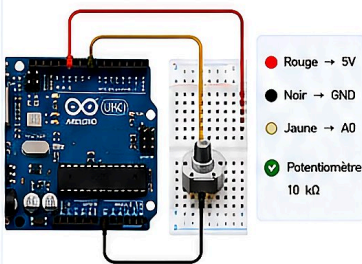
4 EXEMPLE : POTENTIOMÈTRE

Le potentiomètre fournit une tension variable entre 0 V et 5 V.



En tournant le bouton, la tension sur A0 change, et la valeur lue aussi (0 à 1023).

5 SCHÉMA DE CONNEXION



Le curseur (broche centrale) du potentiomètre est connecté à A0.

6 FONCTION ARDUINO UTILISÉE

La fonction `analogRead()` lit la valeur analogique sur la broche spécifiée.

```
int valeur = analogRead(A0);
// Lit la broche A0 (0 à 1023)
```

La valeur retournée est un entier entre 0 et 1023.

`analogRead()` ne prend que les broches A0 à A5 sur Arduino UNO.

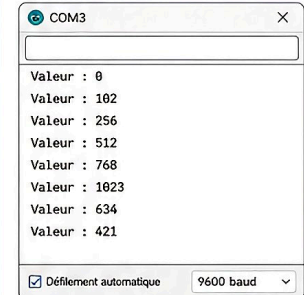
7 EXEMPLE DE CODE

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  int valeur = analogRead(A0); // Lit A0
  Serial.print("Valeur : ");
  Serial.println(valeur);
  delay(500);
}
```

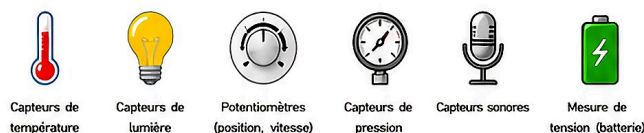
Ouvrez le Moniteur Série à 9600 bauds pour voir les valeurs en temps réel.

8 MONITEUR SÉRIE (EXEMPLE)



Les valeurs changent selon la position du potentiomètre.

9 UTILISATIONS COURANTES



Les entrées analogiques sont essentielles pour mesurer le monde réel avec précision.

10 À RETENIR

- Les entrées analogiques lisent des tensions variables.
- Arduino UNO : 6 entrées (A0 à A5) avec résolution 10 bits (0 à 1023).
- Plage de tension : 0 V à 5 V (par défaut).
- Fonction utilisée : `analogRead(broche)`.
- Très utiles pour les capteurs, potentiomètres et mesures réelles.
- Indirectement, c'est la base de tout système de mesure en robotique.



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

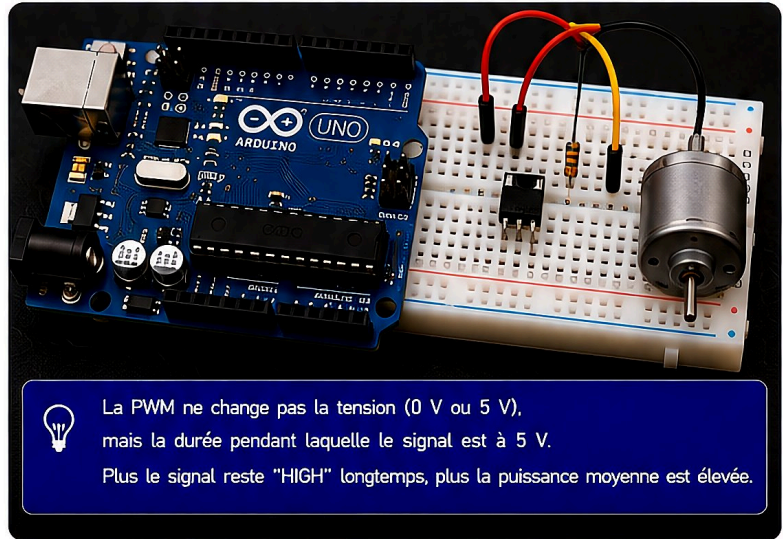
3.6

SORTIES PWM (MODULATION DE LARGEUR D'IMPULSION)

Contrôler la puissance moyenne
avec Arduino et ESP32

La PWM (Pulse Width Modulation) consiste à envoyer une série d'impulsions numériques (ON/OFF) à haute fréquence. En variant la durée de l'état HIGH par rapport à la période, on contrôle la puissance moyenne délivrée à une charge.

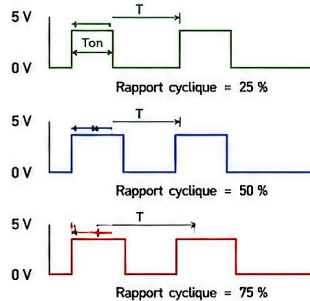
Elle est idéale pour faire varier la vitesse d'un moteur, l'intensité d'une LED, la luminosité d'un écran ou d'un chauffage, etc.



La PWM ne change pas la tension (0 V ou 5 V), mais la durée pendant laquelle le signal est à 5 V. Plus le signal reste "HIGH" longtemps, plus la puissance moyenne est élevée.

1 QU'EST-CE QUE LA PWM ?

La PWM envoie un signal carré rapide entre 0 V (LOW) et 5 V (HIGH). En changeant le rapport cyclique (duty cycle), on varie la puissance moyenne.



Ton = temps à l'état HIGH
T = période du signal
Rapport cyclique (%) = (Ton / T) × 100

2 EXEMPLES DE RAPPORT CYCLIQUE

Rapport cyclique	Signal PWM	Puissance moyenne
0 %		0 % (OFF)
25 %		25 %
50 %		50 %
75 %		75 %
100 %		100 % (ON)



À 0 % la charge est éteinte, à 100 % elle est à pleine puissance.

3 BROCHES PWM SUR ARDUINO UNO

Les broches marquées d'un ~ peuvent générer un signal PWM avec la fonction `analogWrite()`.

Broches PWM (UNO)	
3	~
5	~
6	~
9	~
10	~
11	~



Les autres broches numériques ne peuvent pas générer de PWM.

4 APPLICATIONS COURANTES



Contrôle de vitesse des moteurs DC



Contrôle de luminosité des LED



Régulation de la puissance (chauffage, ventilateurs)



Génération de sons et d'effets audio

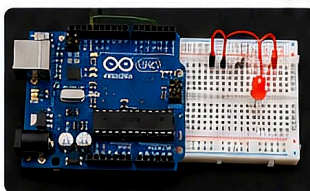
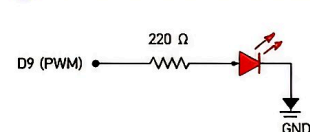


Contrôle de rétroéclairage d'écrans et modules



La PWM permet un contrôle efficace et précis avec très peu de composants.

5 SCHÉMA : VARIATION LUMINOSITÉ LED



En changeant le rapport cyclique PWM, la LED devient plus ou moins lumineuse.

6 CODE EXEMPLE : FADE LED (FONDU)

```
// Variation de luminosité d'une LED
// connectée sur la broche 9 (PWM)
const int led = 9;

void setup() {
  pinMode(led, OUTPUT);
}

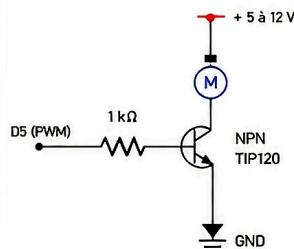
void loop() {
  // Augmente la luminosité
  for (int i = 0; i <= 255; i++) {
    analogWrite(led, i);
    delay(10);
  }
  // Diminue la luminosité
  for (int i = 255; i >= 0; i--) {
    analogWrite(led, i);
    delay(10);
  }
}
```



La valeur de `analogWrite()` varie de 0 à 255 (soit 0 à 100 %).

7 PWM ET MOTEUR DC

Avec un transistor ou un driver, la PWM permet de contrôler la vitesse d'un moteur DC.



Le transistor agit comme un interrupteur rapide commandé par le signal PWM.

8 FRÉQUENCE PWM

La fréquence par défaut sur Arduino UNO est environ 490 Hz (broches 5, 6, 9, 10, 11) et 980 Hz (broches 3 et 11).

Fréquence PWM par défaut	
Broches	Fréquence (approx.)
5, 6, 9, 10	490 Hz
3, 11	980 Hz

Pour la plupart des applications (LED, moteurs), ces fréquences sont parfaites et inaudibles pour l'oreille humaine.



Pour des applications sensibles (audio), on peut modifier la fréquence.

9 AVANTAGES DE LA PWM



Efficacité énergétique
Moins de pertes de puissance



Peu d'échauffement
Le composant commute, il dissipe peu de chaleur



Contrôle précis
De 0 à 100 % avec une résolution fine (0 à 255)



Peu de composants
Simple à réaliser avec Arduino



Silencieux
Aucune gêne sonore pour LED et moteurs à haute fréquence



La PWM est une technique essentielle pour tout projet électronique ou robotique.

10 À RETENIR

- ✓ La PWM envoie un signal carré entre 0 V et 5 V.
- ✓ On contrôle la puissance moyenne en modifiant le rapport cyclique.
- ✓ Arduino UNO possède 6 broches PWM : 3, 5, 6, 9, 10 et 11.
- ✓ La fonction `analogWrite(pin, valeur)` utilise des valeurs de 0 à 255.
- ✓ Applications : LED, moteurs DC, éclairage, chauffage, son, écrans, etc.
- ✓ La PWM est simple, efficace et très utilisée en robotique.



3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

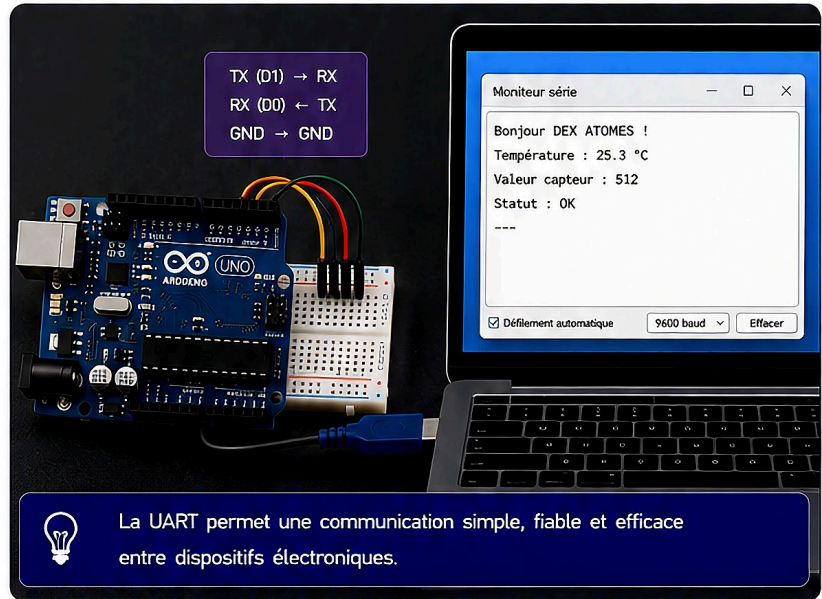
3.7

COMMUNICATION SÉRIE UART

Échanger des données entre cartes et appareils

La communication série UART (Universal Asynchronous Receiver Transmitter) permet d'envoyer et de recevoir des données bit par bit sur seulement deux fils : TX (Transmission) et RX (Réception), plus une masse commune (GND).

Elle est largement utilisée pour communiquer avec des modules (Bluetooth, GPS, capteurs, ESP32), des ordinateurs ou d'autres microcontrôleurs.

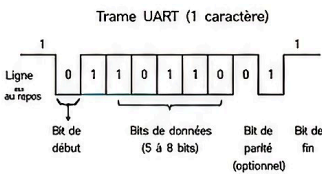


La UART permet une communication simple, fiable et efficace entre dispositifs électroniques.

1 QU'EST-CE QUE L'UART ?

L'UART est un protocole de communication série asynchrone.

- Asynchrone : pas d'horloge partagée.
- Chaque caractère est encadré par des bits de début et de fin.

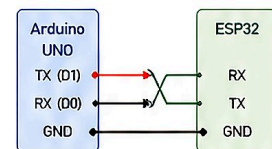


Une trame UART commence toujours par un bit de début (0) et se termine par un bit de fin (1).

2 CONNEXION DE BASE

Pour communiquer entre deux dispositifs, il faut connecter :

- TX de l'émetteur → RX du récepteur
- RX de l'émetteur ← TX du récepteur
- GND commun



Le GND commun est indispensable pour que les niveaux logiques soient identiques.

3 PARAMÈTRES DE COMMUNICATION

Les deux dispositifs doivent avoir les mêmes paramètres :

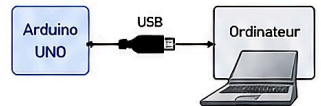
- Débit en bauds (baud rate)
- Nombre de bits de données
- Parité (None, Even, Odd)
- Bits d'arrêt (1 ou 2)

Paramètres courants	
Débit (baud)	9600 (par défaut)
Bits de données	8
Parité	Aucune (None)
Bits d'arrêt	1

Si les paramètres ne sont pas identiques, la communication ne fonctionnera pas.

4 COMMUNICATION AVEC L'ORDINATEUR

La carte peut communiquer avec le PC via le port USB.



Moniteur Série (Arduino IDE)
Outil pratique pour envoyer et recevoir des données en temps réel.

Ouvrez le Moniteur Série et réglez le même débit en bauds que dans votre code.

5 FONCTIONS DE BASE (ARDUINO)

Les fonctions principales de la bibliothèque Serial :

```
Serial.begin(baud); // Initialise la liaison série
Serial.print(data); // Envoie des données
Serial.println(data); // Envoie + retour à la ligne
Serial.available(); // Nombre d'octets reçus
Serial.read(); // Lit un octet
Serial.readString(); // Lit une chaîne de caractères
```

Serial.begin() doit être appelé dans setup() avec le même débit en bauds que le Moniteur Série ou l'autre dispositif.

6 EXEMPLE : ENVOYER UN MESSAGE

Le code ci-dessous envoie un message au Moniteur Série toutes les secondes.

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.println("Bonjour DEX ATOMES !");
  delay(1000); // Attendre 1 seconde
}
```

Ouvrez le Moniteur Série à 9600 bauds : vous verrez le message s'afficher.

7 EXEMPLE : LIRE DES DONNÉES REÇUES

Ce code lit les données reçues depuis le Moniteur Série et les affiche sur la carte.

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  if (Serial.available()) {
    String msg = Serial.readStringUntil('\n');
    Serial.print("Reçu : ");
    Serial.println(msg);
  }
}
```

Tapez un texte dans le Moniteur Série, il sera affiché précédé de "Reçu :".

8 APPLICATIONS COURANTES

- Modules Bluetooth (HC-05, HC-06)
Communication sans fil
- Modules GPS
Lecture de coordonnées
- Capteurs et modules divers
Température, humidité, distance...
- Communication entre microcontrôleurs
Arduino ↔ ESP32, ESP8266, etc.

L'UART est la base de la plupart des systèmes embarqués modernes.

9 DÉBIT EN BAUD : QUELQUES VALEURS COURANTES

Le débit en bauds représente le nombre de bits transmis par seconde. Plus le débit est élevé, plus la transmission est rapide, mais plus elle est sensible aux erreurs.

Baud rate (bps)	1200	2400	4800	9600	19200	38400	57600	115200
Utilisation typique	Anciens modules	Capteurs simples	GPS	Par défaut Arduino	Communication rapide	Modules modernes	Bluetooth rapide	Très rapide

9600 bauds est un bon point de départ pour la majorité des projets.

10 À RETENIR

- ✓ UART utilise seulement TX, RX et GND.
- ✓ Les deux dispositifs doivent avoir les mêmes paramètres.
- ✓ Elle permet d'échanger des données de manière simple et efficace.
- ✓ Utilisez le Moniteur Série pour tester et déboguer.
- ✓ Indispensable pour de nombreux modules et systèmes embarqués.
- ✓ Maîtriser l'UART vous ouvrira la porte à beaucoup de projets avancés !



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

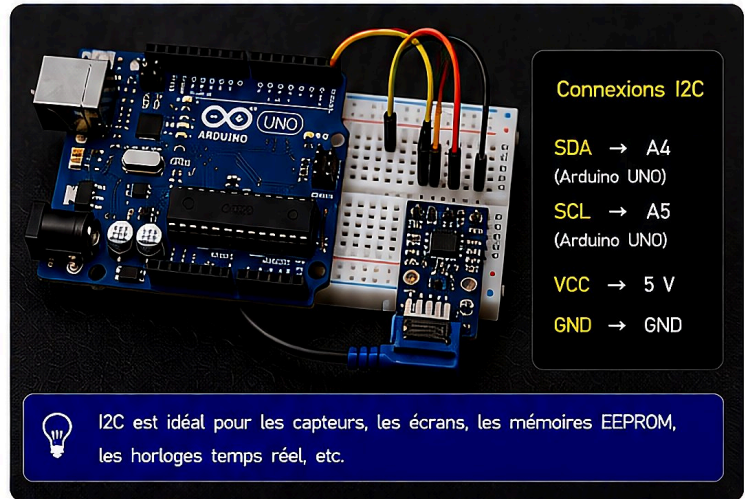
3.8

I2C (INTER-INTÉGRÉ) COMMUNICATION SÉRIE À DEUX FILS

Communiquer avec des capteurs et modules avec Arduino et ESP32

I2C (Inter-Integrated Circuit) est un protocole de communication série synchrone qui utilise seulement deux fils : SDA (données) et SCL (horloge).

Il permet de connecter plusieurs dispositifs sur le même bus grâce à des adresses uniques.



Connexions I2C

SDA → A4
(Arduino UNO)

SCL → A5
(Arduino UNO)

VCC → 5 V

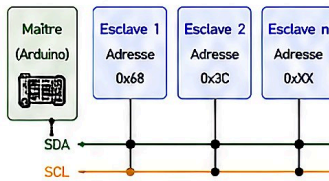
GND → GND



I2C est idéal pour les capteurs, les écrans, les mémoires EEPROM, les horloges temps réel, etc.

1 QU'EST-CE QUE I2C ?

I2C est un protocole série qui permet à plusieurs périphériques (maîtres et esclaves) de communiquer sur seulement deux fils.



Chaque esclave possède une adresse unique (7 bits) pour être identifié sur le bus.

2 LIGNES I2C

Le bus I2C utilise deux lignes partagées :

- SDA (Serial Data) : ligne de données
- SCL (Serial Clock) : ligne d'horloge

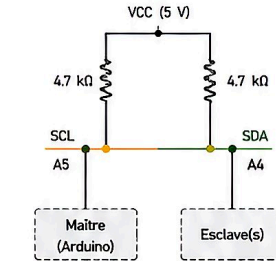


État des lignes	Signification
SDA = 0	Données à 0 (tirée à LOW)
SDA = 1	Données à 1 (tirée à HIGH)
SCL = 0	Horloge à 0
SCL = 1	Horloge à 1

Les lignes sont de type open-drain et nécessitent des résistances de tirage (pull-up).

3 RÉSISTANCES DE TIRAGE (PULL-UP)

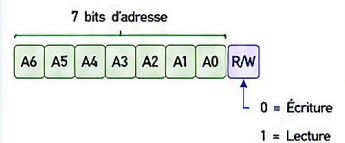
Les résistances pull-up maintiennent les lignes SDA et SCL à l'état HIGH lorsqu'aucun périphérique ne les tire à LOW.



Valeur typique : 4.7 kΩ (entre 2.2 kΩ et 10 kΩ selon le bus et la longueur des fils).

4 ADRESSAGE I2C

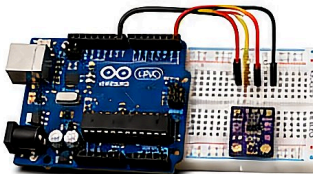
Chaque esclave possède une adresse sur 7 bits. L'adresse sur 8 bits utilisée sur le bus inclut le bit R/W (Lecture/Écriture).



Exemple : Adresse 0x68		
7 bits adresse	8 bits (écriture)	8 bits (lecture)
110 1000	110 1000 0	110 1000 1
0x68	0xD0	0xD1

Vérifiez toujours l'adresse de votre module dans la documentation du fabricant.

5 EXEMPLE DE CONNEXION : CAPTEUR TEMPÉRATURE BMP280



BMP280	→	Arduino UNO
VCC	→	5V
GND	→	GND
SDA	→	A4 (SDA)
SCL	→	A5 (SCL)

De nombreux modules I2C utilisent déjà des pull-up intégrés.

6 FONCTIONS I2C (ARDUINO)

La bibliothèque Wire.h facilite l'utilisation d'I2C.

```
Wire.begin();
Wire.beginTransmission(address);
Wire.write(data);
Wire.endTransmission();
Wire.requestFrom(address, quantity);
Wire.available();
Wire.read();
```

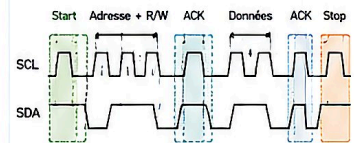
N'oubliez pas d'inclure la bibliothèque :
#include <Wire.h>

7 EXEMPLE : LIRE L'ID DU BMP280

```
#include <Wire.h>
#define BMP280_ADDR 0x76 // Adresse I2C typique
void setup() {
  Serial.begin(9600);
  Wire.begin();
  Serial.println("Recherche BMP280...");
  Wire.beginTransmission(BMP280_ADDR);
  byte error = Wire.endTransmission();
  if (error == 0) {
    Serial.println("BMP280 trouvé !");
  } else {
    Serial.println("BMP280 non trouvé.");
  }
}
```

Beaucoup de capteurs I2C ont plusieurs adresses possibles (ex : 0x76 ou 0x77 pour BMP280).

8 SÉQUENCE DE COMMUNICATION I2C



- Start : le maître initie la communication.
- Adresse : le maître envoie l'adresse + R/W.
- ACK : l'esclave répond (ACK = 0).
- Données : envoi ou réception des données.
- Stop : le maître termine la communication.

Le timing est géré automatiquement par la bibliothèque Wire.

9 DISPOSITIFS I2C COURANTS

Capteurs température (BMP280, DS18B20)	Horloges temps réel (RTC) (DS3231, PCF8523)	Écrans LCD OLED (16x2, OLED)	Mémoires EEPROM (24C02, 24LC256)	Capteurs divers (IMU, MPU6050, VLS3L0X, etc.)

I2C est largement utilisé dans les projets de robotique, domotique, IoT et systèmes embarqués.

10 À RETENIR

- ✓ I2C utilise seulement deux fils : SDA et SCL.
- ✓ Plusieurs dispositifs peuvent partager le même bus.
- ✓ Chaque esclave possède une adresse unique (7 bits).
- ✓ Nécessite des résistances pull-up sur SDA et SCL.
- ✓ La bibliothèque Wire rend la communication très simple.
- ✓ I2C est parfait pour connecter des capteurs et modules avancés.



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

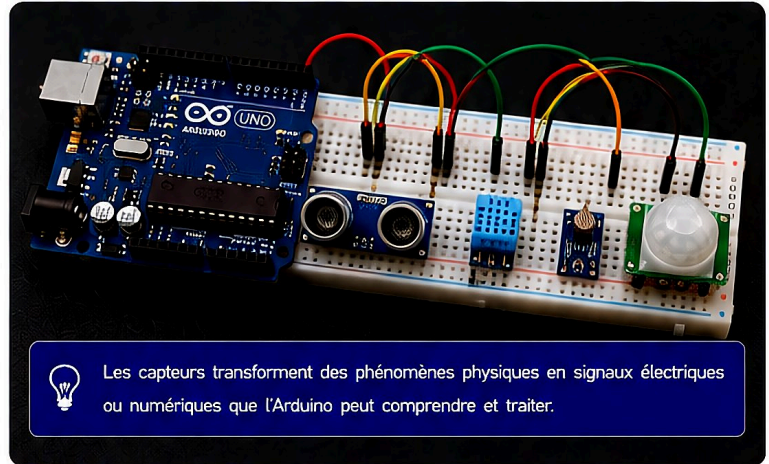
3.9

CAPTEURS ET INTERFACE AVEC ARDUINO

Utiliser des capteurs pour interagir avec le monde réel

Les capteurs permettent à un microcontrôleur de mesurer des grandeurs physiques telles que la température, la lumière, la distance, le son, le mouvement, etc.

Les données lues peuvent ensuite être traitées, affichées ou utilisées pour prendre des décisions dans nos projets.



Les capteurs transforment des phénomènes physiques en signaux électriques ou numériques que l'Arduino peut comprendre et traiter.

1 QU'EST-CE QU'UN CAPTEUR ?

Un capteur détecte une variation dans son environnement et la convertit en un signal exploitable par l'Arduino.

Exemples :

- Température
- Lumière
- Distance
- Mouvement
- Son
- Humidité



Chaque capteur a ses caractéristiques et nécessite une méthode de connexion spécifique.

2 TYPES DE SORTIE DES CAPTEURS

Les capteurs peuvent fournir :

- Signal analogique (tension variable)
- Signal numérique (ON/OFF)
- Signal numérique via protocole (I2C, UART, OneWire, etc.)

Type de signal	Exemple de capteur
Analogique	LM35, LDR, Potentiomètre
Numérique	Bouton, PIR, Fin de course
I2C	BMP280, OLED, RTC DS2231
UART	GPS NEO-6M, Modules Bluetooth
OneWire	DS18B20 (température)



Vérifiez toujours le type de sortie du capteur avant de le connecter.

3 ALIMENTATION DES CAPTEURS

La plupart des capteurs fonctionnent en :

- 3.3 V ou 5 V
- GND commun avec l'Arduino

Capteurs 5 V (compatibles Arduino UNO)



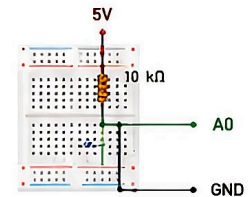
Capteurs 3.3 V (idéal avec ESP32)



Ne jamais appliquer une tension supérieure à celle supportée par le capteur.

4 CONNEXION DE BASE

Exemple d'un capteur analogique (LDR) sur Arduino UNO.



Connexions :

- LDR entre 5V et A0
- Résistance 10 kΩ entre A0 et GND
- Lire la tension sur A0 (0 à 1023)

5 EXEMPLE : LECTURE D'UN LDR

Le code lit la valeur de lumière et l'affiche dans le Moniteur Série.

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  int valeur = analogRead(A0);
  Serial.print("Lumière : ");
  Serial.println(valeur);
  delay(500);
}
```

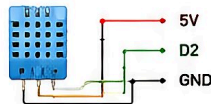


La valeur retournée est entre 0 (obscurité) et 1023 (lumière intense).

6 EXEMPLE : CAPTEUR DE TEMPÉRATURE (DHT11)

Le DHT11 mesure la température et l'humidité.

Connexion : VCC → 5V, Data → D2, GND → GND



```
#include <DHT.h>
#define DHTPIN 2
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  Serial.print("T: "); Serial.print(t);
  Serial.print(" C, H: "); Serial.print(h);
  Serial.println(" %");
  delay(2000);
}
```

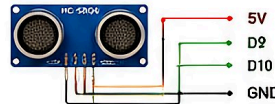


Les capteurs de température et d'humidité sont utilisés dans la domotique et l'agriculture intelligente.

7 EXEMPLE : CAPTEUR DE DISTANCE (HC-SR04)

Mesure la distance par ultrasons.

Connexion : Trig → 99, Echo → D10



```
long duree;
float distance;

void setup() {
  pinMode(9, OUTPUT); pinMode(10, INPUT);
  Serial.begin(9600);
}

void loop() {
  digitalWrite(9, LOW); delayMicroseconds(2);
  digitalWrite(9, HIGH); delayMicroseconds(10);
  digitalWrite(9, LOW);
  duree = pulseIn(10, HIGH);
  distance = duree * 0.034 / 2;
  Serial.print("Distance: ");
  Serial.println(distance);
  delay(500);
}
```



La vitesse du son est d'environ 343 m/s (0.034 cm/µs).

8 CONSEILS PRATIQUES



Vérifiez toujours la tension d'alimentation du capteur.



Reliez toujours les GND entre capteur et Arduino.



Utilisez des résistances de tirage adaptées (pull-up ou pull-down).



Filterez les valeurs pour éviter les lectures instables.



Consultez la documentation du capteur avant utilisation.

9 TABLEAU RÉCAPITULATIF DES CAPTEURS COURANTS

Capteur	Type	Sortie	Plage / Précision	Usage principal
LDR	Lumière	Analogique	0 à 1023	Éclairage automatique
DHT11	Température / Humidité	Numérique	0-50°C, ±2°C	Environnement
HC-SR04	Distance	Numérique	2 à 400 cm	Évitement d'obstacles
PIR HC-SR501	Mouvement	Numérique	3 à 7 m	Sécurité, détection
BMP280	Pression / Temp.	I2C	±1hPa, ±1.0°C	Météo, altitude
DS18B20	Température	OneWire	-55 à 125°C	Mesure précise
MPU6050	Accéléro / Gyroscope	I2C	±2g / ±250°/s	Robotique, drones

10 À RETENIR

- ✓ Les capteurs permettent à l'Arduino d'interagir avec le monde réel.
- ✓ Il existe différents types de sorties : analogique, numérique, I2C, UART, OneWire.
- ✓ Respectez la tension d'alimentation et connectez toujours les GND.
- ✓ Utilisez le bon protocole et les bibliothèques adaptées.
- ✓ Testez toujours vos capteurs avec le Moniteur Série.
- ✓ La combinaison de plusieurs capteurs rend vos projets plus intelligents !



3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

3.10

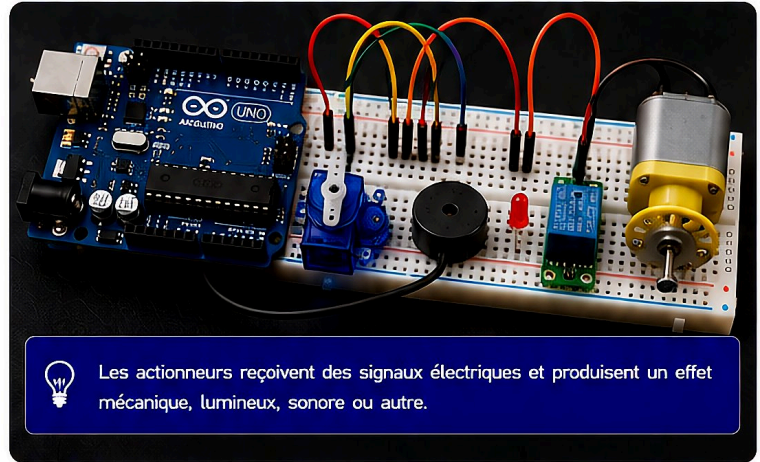
ACTIONNEURS ET SORTIES AVANCÉES

Contrôler le monde physique avec Arduino et ESP32

Les actionneurs sont des dispositifs qui exécutent des actions physiques en réponse aux commandes d'un microcontrôleur.

Exemples : LED, moteurs, servomoteurs, relais, buzzer, écran LCD, etc.

Cette section montre comment les contrôler et les interfacer avec Arduino et ESP32.



Les actionneurs reçoivent des signaux électriques et produisent un effet mécanique, lumineux, sonore ou autre.

1 QU'EST-CE QU'UN ACTIONNEUR ?

Un actionneur transforme une commande électrique en action physique.

Exemples courants :

- LED : lumière
- Moteur DC : mouvement
- Servo : position précise
- Buzzer : son
- Relais : contrôle de charges
- Écran LCD : affichage



Les actionneurs sont utilisés pour interagir avec l'environnement.

2 TYPES DE COMMANDE

Les actionneurs peuvent être commandés par :

- Signaux numériques (ON/OFF)
- PWM (Pulse Width Modulation)
- Communication (I2C, SPI, UART)
- Niveau analogique

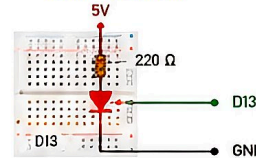
Type de commande	Utilisation typique
Numérique	LED, Relais, Buzzer
PWM	Moteurs, Servos, LED dimmable
Analogique	Variateurs, DAC externes
I2C	Écrans LCD, Drivers, Modules
SPI/UART	Modules spécialisés

Choisissez le type de commande adapté à votre actionneur.

3 COMMANDE D'UNE LED (NUMÉRIQUE)

La LED s'allume quand la broche est à HIGH. Elle s'éteint quand la broche est à LOW.

Schéma de connexion



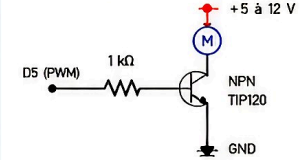
```
void setup() {
  pinMode(13, OUTPUT);
}
void loop() {
  digitalWrite(13, HIGH); // LED ON
  delay(1000);
  digitalWrite(13, LOW); // LED OFF
  delay(1000);
}
```

Utilisez toujours une résistance avec la LED pour limiter le courant (220 Ω typique).

4 COMMANDE D'UN MOTEUR DC (PWM)

Le PWM permet de contrôler la vitesse d'un moteur DC.

Schéma de connexion (avec transistor)



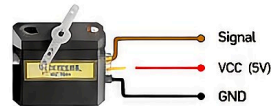
```
void setup() { pinMode(5, OUTPUT); }
void loop() {
  for (int v = 0; v <= 255; v++) {
    analogWrite(5, v);
    delay(20);
  }
}
```

analogWrite() accepte des valeurs de 0 à 255. Plus la valeur est grande, plus le moteur va vite.

5 SERVOMOTEUR (PWM)

Le servomoteur permet un contrôle précis de position (généralement 0° à 180°).

Schéma de connexion



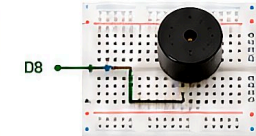
```
#include <Servo.h>
Servo monServo;
void setup() {
  monServo.attach(9); // Signal sur D9
}
void loop() {
  monServo.write(0); // 0°
  delay(1000);
  monServo.write(90); // 90°
  delay(1000);
  monServo.write(180); // 180°
  delay(1000);
}
```

Certains servos nécessitent une alimentation externe (5V) si le courant est élevé.

6 BUZZER ACTIF

Le buzzer actif émet un son lorsqu'il est alimenté en HIGH.

Schéma de connexion



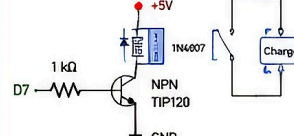
```
void setup() { pinMode(8, OUTPUT); }
void loop() {
  digitalWrite(8, HIGH); // Son ON
  delay(500);
  digitalWrite(8, LOW); // Son OFF
  delay(500);
}
```

Pour changer la fréquence ou jouer une note, utilisez la fonction tone().

7 RELAIS (CONTRÔLE DE CHARGE)

Le relais permet de commuter des appareils à haute tension avec un signal logique.

Schéma de connexion



```
void setup() { pinMode(7, OUTPUT); }
void loop() {
  digitalWrite(7, HIGH); // Relais ON
  delay(2000);
  digitalWrite(7, LOW); // Relais OFF
  delay(2000);
}
```

La diode de roue libre (1N4007) protège le transistor des pics de tension.

8 ÉCRAN LCD 16x2 (I2C)

L'écran LCD permet d'afficher du texte. Le module I2C utilise seulement 2 broches.

Connexions

LCD I2C	Arduino UNO
VCC	5V
GND	GND
SDA	A4
SCL	A5



Bibliothèque : LiquidCrystal_I2C
Adresse par défaut : 0x27 ou 0x3F (varie selon le module).

9 RÉCAPITULATIF DES ACTIONNEURS

Actionneur	Type de commande	Broches typiques	Utilisation
LED	Numérique	D2 – D13	Indication visuelle
Moteur DC	PWM	D3, D5, D6, D9, D10, D11	Contrôle de vitesse
Servomoteur	PWM	D3, D5, D6, D9, D10, D11	Position précise
Buzzer actif	Numérique / tone()	D2 – D13	Alarmes sonores
Relais	Numérique	D2 – D13	Contrôle de charges
Écran LCD (I2C)	I2C	A4 (SDA), A5 (SCL)	Affichage de texte

Combinez plusieurs actionneurs pour créer des systèmes interactifs et utiles !

10 À RETENIR

- ✓ Les actionneurs exécutent des actions physiques.
- ✓ Les LEDs sont contrôlées en numérique.
- ✓ Les moteurs et servos utilisent le PWM.
- ✓ Les buzzers produisent des sons.
- ✓ Les relais permettent de contrôler des charges externes.
- ✓ Les écrans LCD affichent des informations à l'utilisateur.
- ✓ Bien choisir le type de commande et la connexion est essentiel pour un système fiable et efficace.



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

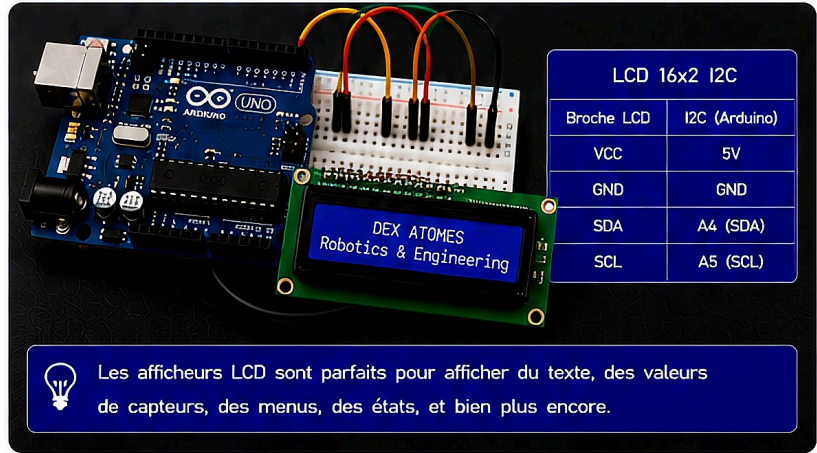
3.11

AFFICHEURS ET ÉCRANS LCD

Afficher des informations sur vos projets

Les afficheurs permettent de communiquer des informations à l'utilisateur.

Le plus courant avec Arduino est l'écran LCD 16x2, mais il existe aussi des afficheurs OLED, 7 segments, et des écrans couleur.



Les afficheurs LCD sont parfaits pour afficher du texte, des valeurs de capteurs, des menus, des états, et bien plus encore.

1 TYPES D'AFFICHEURS

Il existe plusieurs types d'afficheurs compatibles avec Arduino et ESP32.

- LCD 16x2 (caractères)
- OLED (I2C, SPI)
- Écrans 7 segments
- Écrans TFT couleur
- Matrices LED (MAX7219)
- Écrans e-paper
- Nextion (tactile)



Chaque afficheur a ses avantages selon le besoin, la taille et la consommation.

2 ÉCRAN LCD 16x2

Le LCD 16x2 peut afficher 16 caractères sur 2 lignes.



Caractéristiques

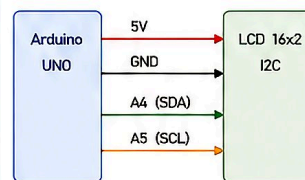
- 16 caractères par ligne
- 2 lignes
- Rétroéclairage (LED)
- Faible consommation
- Très utilisé dans les projets



Idéal pour les menus, les données de capteurs, les messages et l'interface.

3 CONNEXION EN MODE I2C (RECOMMANDÉ)

Le module LCD avec I2C ne nécessite que 4 connexions.



Adresse I2C par défaut : 0x27 ou 0x3F



Le mode I2C économise des broches et simplifie le câblage.

4 BIBLIOTHÈQUE ET INITIALISATION

Nous utilisons la bibliothèque LiquidCrystal_I2C.

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

LiquidCrystal_I2C lcd(0x27, 16, 2);
// Adresse, colonnes, lignes

void setup() {
  lcd.init(); // Initialise l'écran
  lcd.backlight(); // Allume le rétroéclairage
  lcd.clear(); // Efface l'écran
  lcd.setCursor(0, 0);
  lcd.print("DEX ATOMES");
  lcd.setCursor(0, 1);
  lcd.print("Bienvenue !");
}
```



Vérifiez l'adresse I2C avec un scanner (voir code à la page suivante).

5 AFFICHER DU TEXTE

Fonctions principales :

- `setCursor(col, ligne)` : position du curseur
- `print()` : affiche du texte
- `clear()` : efface l'écran
- `backlight()` / `noBacklight()`

```
lcd.setCursor(0, 0);
lcd.print("Température :");
lcd.setCursor(0, 1);
lcd.print("25.3 °C");
```



Le curseur commence à (0,0) en haut à gauche.

6 AFFICHER DES NOMBRES ET VARIABLES

Utilisez `print()` pour afficher des valeurs numériques.

```
int temp = 25;
float volt = 3.75;

lcd.setCursor(0, 0);
lcd.print("Temp: ");
lcd.print(temp);
lcd.print((char)223); // Symbole °
lcd.print("C");

lcd.setCursor(0, 1);
lcd.print("Volt: ");
lcd.print(volt, 2); // 2 décimales
lcd.print(" V");
```



Utilisez `print()` pour enchaîner texte et valeurs facilement.

7 DÉFILEMENT DU TEXTE

Pour afficher un texte plus long que 16 caractères.

```
const char message[] = "DEX ATOMES Robotics & Engineering";

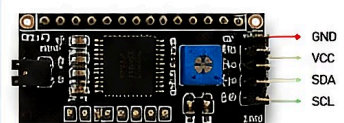
void loop() {
  for (int i = 0; i < 28; i++) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(message.substring(i));
    delay(250);
  }
}
```



Le texte défile de la droite vers la gauche.

8 EFFETS UTILES

- Clignoter le curseur
- Effacer une ligne
- Afficher un caractère personnalisé
- Ajuster le contraste (via potentiomètre sur le module I2C)



Le potentiomètre bleu ajuste le contraste de l'écran.

9 EXEMPLES DE PROJETS AVEC LCD

Station météo	Compteur de visiteurs	Système de menu	Horloge numérique
Affiche température et humidité.	Incréméte et affiche le nombre.	Navigation simple avec boutons.	Affiche l'heure et la date.



Les afficheurs rendent vos projets plus interactifs et professionnels.

10 À RETENIR

- ✓ Le LCD 16x2 est simple, fiable et très utilisé.
- ✓ Le mode I2C réduit le câblage à seulement 4 fils.
- ✓ La bibliothèque `LiquidCrystal_I2C` facilite la programmation.
- ✓ Vous pouvez afficher du texte, des nombres, des menus et des données.
- ✓ Les afficheurs améliorent l'expérience utilisateur de vos projets.
- ✓ Pratiquez l'affichage de valeurs réelles dans vos projets.



BONUS : SCANNEZ L'ADRESSE I2C

Le code suivant scanne le bus I2C pour trouver l'adresse de votre module LCD.

```
#include <Wire.h>

void setup() {
  Wire.begin(); Serial.begin(9600);
  Serial.println("Scan I2C...");
}

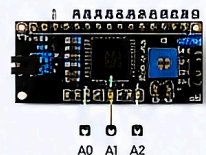
void loop() {
  byte error, address;
  int nDevices = 0;
  for (address = 1; address < 127; address++) {
    Wire.beginTransmission(address);
    error = Wire.endTransmission(1);
    if (error == 0) {
      Serial.print("Adresse trouvée : 0x");
      Serial.print(address, HEX);
      Serial.println();
      nDevices++;
    }
    if (nDevices == 0) Serial.println("Aucun appareil trouvé"); delay(5000);
  }
}
```



Adresse par défaut : 0x27 ou 0x3F

Changez l'adresse si nécessaire avec les cavaliers A0, A1, A2.

A0	A1	A2	Don
0	0	0	0x27
1	1	1	0x3F



3

CHAPITRE 3 Arduino et ESP32 – Fondamentaux

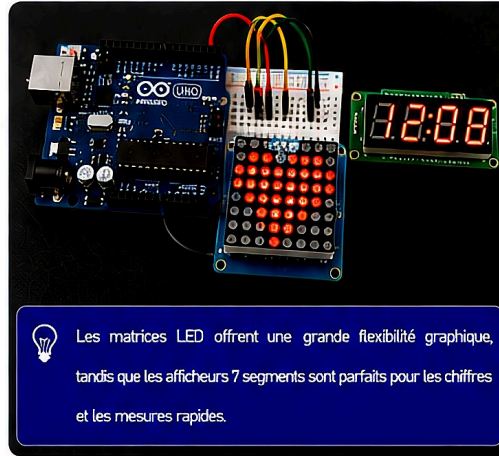
3.12

MATRICES LED ET AFFICHEURS 7 SEGMENTS

Créer des affichages dynamiques
et des interfaces visuelles

Les matrices LED et les afficheurs 7 segments permettent d'afficher des textes, des icônes, des animations ou des nombres de manière claire et efficace.

Ils sont très utilisés dans les projets électroniques, la signalisation et les interfaces utilisateur.



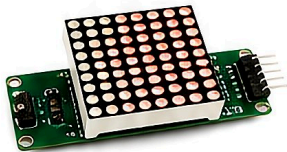
Les matrices LED offrent une grande flexibilité graphique, tandis que les afficheurs 7 segments sont parfaits pour les chiffres et les mesures rapides.

TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	70
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73

1 MATRICE LED 8x8 (MAX7219)

Module matrice 8x8 contrôlé par le driver MAX7219 via 3 fils (SPI).



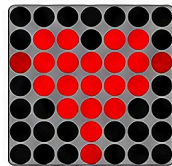
Connexions (MAX7219 → Arduino)

VCC	→	5V
GND	→	GND
DIN	→	D11 (MOSI)
CS	→	D10
CLK	→	D13 (SCK)

Le MAX7219 gère le multiplexage, ce qui rend l'utilisation très simple.

2 EXEMPLE : AFFICHER UN CŒUR

Nous affichons un cœur sur la matrice 8x8.



```
#include <HD_MAX72XX.h>
#define CS_PIN 10
HD_MAX72XX mx(HD_MAX72XX::FC16_HW, CS_PIN, 1);

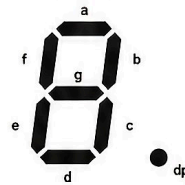
byte coeur[8] = {
  0b00111100,
  0b01111110,
  0b11111111,
  0b11111111,
  0b11111111,
  0b01111110,
  0b00111100,
  0b00000000
};

void setup() { mx.begin(); }
void loop() {
  mx.clear();
  mx.setRow(0,0,coeur[0]);
  mx.setRow(0,1,coeur[1]);
  mx.setRow(0,2,coeur[2]);
  mx.setRow(0,3,coeur[3]);
  mx.setRow(0,4,coeur[4]);
  mx.setRow(0,5,coeur[5]);
  mx.setRow(0,6,coeur[6]);
  mx.setRow(0,7,coeur[7]);
}
```

Les données sont envoyées ligne par ligne.

3 AFFICHEUR 7 SEGMENTS

Chaque chiffre est composé de 7 segments (a à g) + le point décimal (dp).



Segments	Broche typique
a	→ D2
b	→ D3
c	→ D4
d	→ D5
e	→ D6
f	→ D7
g	→ D8
dp	→ D9

Un chiffre est formé en allumant les bons segments.

4 AFFICHEUR 7 SEGMENTS (4 CHIFFRES)

Utilisation fréquente du module TM1637 (2 fils seulement).



Connexions (TM1637 → Arduino)

VCC	→	5V
GND	→	GND
DIO	→	D2
CLK	→	D3

Le TM1637 gère l'affichage et la luminosité.

5 AFFICHER UN NOMBRE (TM1637)

Exemple : afficher la température 25.6 °C.

```
#include <TM1637Display.h>
#define CLK 3
#define DIO 2
TM1637Display display(CLK, DIO);

void setup() {
  display.setBrightness(7);
}

void loop() {
  float temp = 25.6;
  display.showNumberDecEx((int)temp*10,
    0b01000000, true); // 25.6
  delay(1000);
}
```

Le point décimal est indiqué par le paramètre (0b01000000).

6 DÉFILEMENT DE TEXTE SUR MATRICE LED

Affiche un message qui défile horizontalement.

```
const char message[] = "DEX ATOMES !";
uint8_t buffer[8];

void loop() {
  for (int i = 0; i < strlen(message); i++){
    mx.transform(HD_MAX72XX::TSL);
    mx.setChar(7, message[i]);
    delay(120);
  }
}
```

La fonction transform() décale tout le contenu de la matrice.

7 ANIMATION D'UN COMPTEUR

Compteur numérique 0 à 9999 sur afficheur 7 segments.

```
int compteur = 0;

void loop() {
  display.showNumberDec(compteur, true);
  delay(500);
  compteur++;
  if (compteur > 9999) compteur = 0;
}
```

Idéal pour les minuteurs, compteurs, et affichages de mesures.

8 CONSEILS PRATIQUES

- Régler la luminosité pour économiser de l'énergie.
- Utiliser des résistances si les segments sont connectés directement.
- Organiser le code avec des fonctions pour plus de lisibilité.
- Tester chaque composant avant de l'intégrer dans le projet final.
- Consulter la documentation des modules (MAX7219, TM1637, etc.).

9 COMPARAISON RAPIDE

Caractéristique	Matrice LED 8x8 (MAX7219)	Afficheur 7 segments (TM1637)	OLED 0.96" I2C (référence)
Type d'affichage	Graphique (pixels)	Numérique (chiffres)	Texte + Graphique
Résolution	8 x 8	4 chiffres (7 segments)	128 x 64
Fils utilisés	3 (SPI)	2 (I2C-like)	2 (I2C)
Contraste	Bon	Excellent	Excellent
Utilisation	Icônes, animations, barres, jeux	Heure, température, compteurs	Menus, graphiques, informations
Consommation	Moyenne	Faible	Faible

Choisissez l'afficheur adapté à votre besoin pour obtenir le meilleur résultat.

10 À RETENIR

- Les matrices LED permettent des affichages graphiques et des animations.
- Les afficheurs 7 segments sont parfaits pour les nombres et les mesures.
- Le MAX7219 simplifie l'utilisation des matrices LED 8x8.
- Le TM1637 réduit le câblage pour les afficheurs 7 segments.
- Bien gérer la luminosité améliore la lisibilité et la durée de vie.
- Pratiquez avec des projets réels pour maîtriser l'affichage dynamique !



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

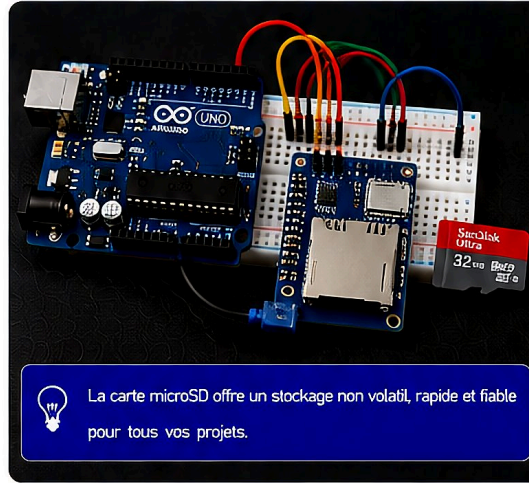
3.13

STOCKAGE ET CARTES MÉMOIRE

Enregistrer et lire des données avec Arduino et ESP32

Les cartes mémoire comme la microSD permettent de stocker des données en grande quantité : journaux, mesures, fichiers texte, images, configurations, etc.

Elles sont essentielles pour les projets autonomes, les enregistreurs de données et les systèmes embarqués.



La carte microSD offre un stockage non volatil, rapide et fiable pour tous vos projets.

TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	70
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73

1 QU'EST-CE QU'UNE CARTE SD ?

Une carte microSD est un support de stockage amovible utilisé pour enregistrer et lire des fichiers.

Caractéristiques courantes :

- Capacité : de 128 Mo à 1 To+
- Interface : SPI (mode le plus utilisé)
- Système de fichiers : FAT16 / FAT32
- Tension : 3.3 V ou 5 V (selon module)



Utilisez toujours des cartes de bonne qualité pour éviter la corruption des données.

2 MODULE LECTEUR MICROSD (SPI)

Le lecteur microSD communique avec Arduino via le bus SPI.

Broche module SD	Arduino UNO
VCC	5V
GND	GND
MOSI	D11
MISO	D12
SCK	D13
CS	D10 (SS)

SPI (mode 0)
CPOL = 0
CPHA = 0

La broche CS (Chip Select) peut être changée, mais doit être définie dans le code.

3 BIBLIOTHÈQUE ARDUINO

Nous utilisons la bibliothèque SD.h incluse dans l'IDE Arduino.

```
#include <SPI.h>
#include <SD.h>
```

Définissez la broche CS utilisée :

```
const int CS_PIN = 10;
```

SD.begin(CS_PIN) initialise la communication avec la carte.

4 INITIALISATION DE LA CARTE SD

Vérifie la présence de la carte et l'initialise.

```
void setup() {
  Serial.begin(9600);
  while(!Serial){}

  if (!SD.begin(CS_PIN)) {
    Serial.println("Échec initialisation SD");
    return;
  }

  Serial.println("Carte SD détectée !");
}
```

Si l'initialisation échoue, vérifiez le câblage, la carte et la broche CS.

5 ÉCRIRE DANS UN FICHIER

Crée un fichier et écrit du texte.

```
File fichier = SD.open("log.txt", FILE_WRITE);
if (fichier) {
  fichier.println("DEX ATOMES");
  fichier.println("Enregistrement de données");
  fichier.println(25.6); // exemple
  fichier.close();
} else {
  Serial.println("Erreur ouverture fichier");
}
```

FILE_WRITE crée le fichier s'il n'existe pas, sinon écrit à la fin.

6 LIRE UN FICHIER

Ouvre un fichier et lit son contenu.

```
File fichier = SD.open("log.txt");
if (fichier) {
  while (fichier.available()) {
    Serial.write(fichier.read());
  }
  fichier.close();
} else {
  Serial.println("Impossible d'ouvrir le fichier");
}
```

available() retourne le nombre d'octets disponibles à lire.

7 EXEMPLE : ENREGISTRER DES MESURES

Enregistre la température et l'humidité toutes les 10 secondes.

```
#include <SPI.h>
#include <SD.h>

const int CS_PIN = 10;

void loop() {
  float temp = 26.3; // Exemple
  float hum = 40.7; // Exemple

  File f = SD.open("donnees.csv", FILE_WRITE);
  if (f) {
    f.print(millis()); f.print(",");
    f.print(temp); f.print(",");
    f.print(hum); f.print(",");
    f.close();
  }
  delay(10000);
}
```

Le format CSV est idéal pour analyser les données sur un ordinateur.

8 COMMANDES UTILES

Commande	Description
SD.exists("fichier.txt")	Vérifie si le fichier existe
SD.remove("fichier.txt")	Supprime un fichier
SD.mkdir("nom")	Crée un dossier
SD.open("f", FILE_READ)	Ouvre pour lecture
SD.open("f", FILE_WRITE)	Ouvre pour écriture
file.size()	Taille du fichier (octets)
file.seek(pos)	Va à la position (octets)
file.close()	Ferme le fichier

Fermez toujours les fichiers après utilisation pour éviter les pertes de données.

9 BONNES PRATIQUES

- ✓ Utilisez des cartes de marque (SanDisk, Kingston, Samsung).
- ✓ Formatez la carte en FAT32 (taille d'unité d'allocation : 32 Ko).
- ✓ Vérifiez toujours l'échec des opérations (ouverture, écriture, lecture).
- ✓ Fermez les fichiers après chaque utilisation.
- ✓ Évitez d'écrire trop fréquemment (use leveling de la carte).
- ✓ Gardez vos noms de fichiers courts et simples (8.3 compatible).
- ✓ Testez la carte sur votre PC avant de l'utiliser dans le projet.



Une bonne gestion du stockage garantit la fiabilité et la durabilité de votre projet.

10 À RETENIR

- ✓ Les cartes microSD permettent un stockage non volatil et amovible.
- ✓ La communication se fait généralement via SPI.
- ✓ La bibliothèque SD.h simplifie la lecture et l'écriture.
- ✓ Idéal pour journaux, enregistreurs de données, fichiers texte, systèmes autonomes.
- ✓ Respectez les bonnes pratiques pour assurer la fiabilité des données.



IDÉE PROJET

Créez un enregistreur de température avec horodatage. Stockez les données sur microSD et analysez-les sur Excel.



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

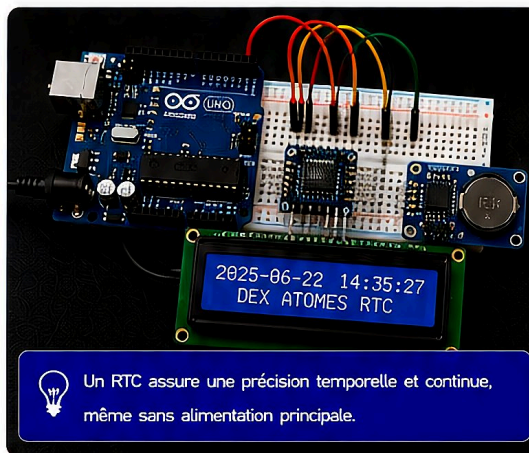
3.14

HORLOGE TEMPS RÉEL (RTC) ET PROJETS AVEC ARDUINO

Gérer la date, l'heure et réaliser des projets concrets

Les modules RTC (Real Time Clock) permettent à Arduino de garder l'heure même lorsqu'il est éteint.

Ils sont essentiels pour les horloges, les enregistreurs de données et les systèmes de planification.



Un RTC assure une précision temporelle et continue, même sans alimentation principale.

TABLE DES MATIÈRES

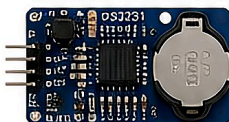
3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	70
3.13	Stockage et cartes mémoire	71
3.13	Stockage temps réel (RTC)	72
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73

1 QU'EST-CE QU'UN RTC ?

Un RTC (Real Time Clock) est une horloge électronique qui fournit l'heure, la date et parfois la température.

Il fonctionne grâce à une pile bouton qui maintient l'horloge en marche.

Exemple populaire : DS3231 (très précis)

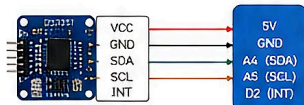


Le DS3231 possède une précision de ± 2 ppm (très élevée) et une compensation de température.

2 MODULE RTC DS3231 (I2C)

Le DS3231 communique avec Arduino via I2C. Adresse I2C par défaut : 0x68

Broche DS3231	Fonction	Connexion Arduino
VCC	Alimentation	5V
GND	Masse	GND
SDA	Données	A4 (SDA)
SCL	Horloge	A5 (SCL)
INT/SQW	Interruption	(Optionnel) D2



Connectez une pile CR2032 pour garder l'heure lorsque l'Arduino est éteint.

3 BIBLIOTHÈQUE RTCLIB

Nous utilisons la bibliothèque RTCLib.h de Adafruit (compatible avec DS3231).

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

```
RTC_DS3231 rtc;
```

Installez la bibliothèque via le gestionnaire de bibliothèques de l'IDE Arduino.

Vérifiez l'adresse I2C avec un scanner si nécessaire (voir page 65).

4 RÉGLER L'HEURE (UNE FOIS)

Ce code permet de régler l'heure dans le RTC.

```
void setup() {
  Serial.begin(9600);
  if (!rtc.begin()) {
    Serial.println("RTC non trouvé");
    while (1);
  }

  // Régler la date et l'heure
  rtc.adjust(DateTime(2025, 6, 22, 14, 30, 0));
  Serial.println("Heure réglée !");
}
```

Commentez ensuite `rtc.adjust(...)` pour ne pas réinitialiser à chaque démarrage.

5 LIRE L'HEURE DU RTC

Ce code lit et affiche la date et l'heure.

```
void loop() {
  DateTime now = rtc.now();
  Serial.print(now.year()); Serial.print("-");
  Serial.print(now.month()); Serial.print("-");
  Serial.print(now.day()); Serial.print(" ");
  Serial.print(now.hour()); Serial.print(":");
  Serial.print(now.minute()); Serial.print(":");
  Serial.println(now.second());
  delay(1000);
}
```

Format affiché : AAAA-MM-JJ HH:MM:SS

La fonction `now()` lit l'heure courante depuis le module RTC.

6 AFFICHAGE SUR LCD 16x2

Affichons l'heure sur un écran LCD 16x2 via I2C (adresse 0x27 ou 0x3F).

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  lcd.init();
  lcd.backlight();
}

void loop() {
  DateTime now = rtc.now();
  lcd.setCursor(0, 0);
  lcd.print(now.year()); lcd.print("-");
  lcd.print(now.month()); lcd.print("-");
  lcd.print(now.day());
  lcd.setCursor(0, 1);
  lcd.print(now.hour()); lcd.print(":");
  lcd.print(now.minute()); lcd.print(":");
  lcd.print(now.second());
  delay(1000);
}
```

Un affichage clair et lisible pour vos projets en temps réel.

7 UTILISATION DE L'ALARME

Le DS3231 possède deux alarmes programmables.

```
void setup() {
  rtc.begin();

  // Alarme 1 : tous les jours à 07:30:00
  rtc.setAlarm1(DateTime(2025, 6, 22, 7, 30, 0),
    DS3231_A1_Day);
  rtc.enableAlarm(1);
  rtc.clearAlarm(1);
}

void loop() {
  if (rtc.alarmFired(1) {
    Serial.println("Alarme 1 déclenchée !");
    rtc.clearAlarm(1);
  }
}
```

Idéal pour réveils, rappels ou enregistrements planifiés.

8 APPLICATIONS COURANTES



Horloges et calendriers



Enregistreurs de données (data loggers)



Alarmes et rappels



Automatisation et domotique



Systèmes de planification et de synchronisation

Les RTC rendent vos projets fiables et autonomes dans le temps.

9 EXEMPLE DE PROJET : HORLOGE NUMÉRIQUE

Objectif : Afficher l'heure sur LCD et clignoter les deux-points chaque seconde.



Fonctionnalités :

- ✓ Affichage sur LCD 16x2
- ✓ Mise à jour chaque seconde
- ✓ Heure persistante avec RTC
- ✓ Simple, fiable et utile

Ce projet peut être amélioré avec un boîtier, des boutons pour régler l'heure, et une batterie de secours.

10 À RETENIR

- ✓ Un RTC permet de garder une heure précise même sans alimentation.
- ✓ Le module DS3231 est précis, stable et facile à utiliser.
- ✓ La communication I2C simplifie le câblage avec seulement 2 fils.
- ✓ Les bibliothèques RTCLib et Wire facilitent le développement.
- ✓ Les alarmes et la planification ouvrent la porte à des projets avancés.
- ✓ Combinez le RTC avec un LCD, une carte SD ou des capteurs pour créer des systèmes intelligents.



BONUS : SAUVEGARDE DES DONNÉES AVEC HORODATAGE

Associez un RTC avec une carte SD pour enregistrer des données datées et horodatées. Exemple : température, humidité, état d'un système, etc.



Exemple de ligne enregistrée :

2025-06-22,14:35:27,25,6,60,3

3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

3.15

PROJETS PRATIQUES AVEC ARDUINO ET ESP32

Appliquer les connaissances pour
créer des systèmes réels

Rien ne vaut la pratique ! Cette section regroupe plusieurs projets concrets qui combinent capteurs, actionneurs, affichage et communication.

Ces exemples vous aideront à consolider vos compétences et à réaliser des systèmes utiles dans la vie quotidienne.

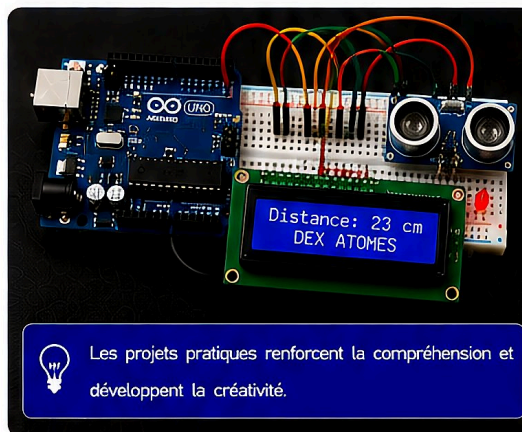


TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73

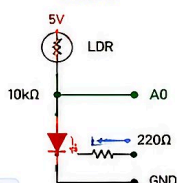
1 PROJET 1 : LAMPE AUTOMATIQUE

Allume une LED automatiquement lorsque la lumière ambiante est faible.

Composants

- LDR (photo-résistance)
- Résistance 10 kΩ
- LED
- Résistance 220 Ω
- Arduino Uno

Schéma



Code

```
int ldrPin = A0;
int ledPin = 13;

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  int valeur = analogRead(ldrPin);
  if (valeur < 500) digitalWrite(ledPin, HIGH);
  else digitalWrite(ledPin, LOW);
  delay(100);
}
```

✓ Résultat : La LED s'allume dans l'obscurité.
Utilisation : Éclairage automatique, veilleuse, sécurité.

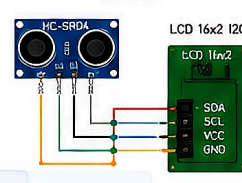
2 PROJET 2 : DÉTECTEUR DE DISTANCE

Mesure la distance avec un capteur ultrason et affiche la valeur sur un LCD.

Composants

- HC-SR04
- LCD 16x2 I2C
- Arduino Uno
- Fils de connexion

Schéma



Code (extrait)

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

LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() { lcd.init(); lcd.backlight(); }

float distance() {
  digitalWrite(9, LOW); delayMicroseconds(2);
  digitalWrite(9, HIGH); delayMicroseconds(10);
  digitalWrite(9, LOW);
  long duration = pulseIn(10, HIGH);
  return duration * 0.034 / 2; // cm
}
```

✓ Résultat : La distance s'affiche en temps réel.
Utilisation : Robotique, stationnement, sécurité.

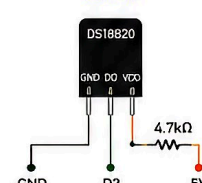
3 PROJET 3 : THERMOMÈTRE NUMÉRIQUE

Lit la température avec un capteur et l'affiche sur un LCD.

Composants

- Capteur DS18B20
- Résistance 4.7 kΩ
- LCD 16x2 I2C
- Arduino Uno

Schéma



Code (extrait)

```
#include <OneWire.h>
#include <DallasTemperature.h>

OneWire oneWire(2);
DallasTemperature sensors(&oneWire);

void setup() {
  lcd.init(); lcd.backlight();
  sensors.begin();
}

float lireTemp() {
  sensors.requestTemperatures();
  return sensors.getTempCByIndex(0);
}
```

✓ Résultat : Température en °C affichée.
Utilisation : Stations météo, contrôle environnemental.

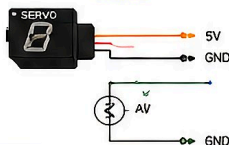
4 PROJET 4 : SERVO MOTEUR CONTRÔLÉ PAR POTENTIOMÈTRE

Fait tourner un servo moteur selon la position d'un potentiomètre.

Composants

- Potentiomètre 10 kΩ
- Servo SG90
- Arduino Uno

Schéma



Code (extrait)

```
#include <Servo.h>
Servo monServo;

void setup() { monServo.attach(9); }

void loop() {
  int val = analogRead(A0);
  int angle = map(val, 0, 1023, 0, 180);
  monServo.write(angle);
  delay(15);
}
```

✓ Résultat : Le servo suit la position du potentiomètre.
Utilisation : Robotique, bras mécanique, commandes.

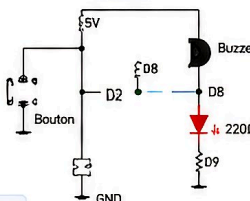
5 PROJET 5 : SYSTÈME D'ALARME SIMPLE

Déclenche un buzzer et une LED lorsqu'un bouton est pressé.

Composants

- Bouton poussoir
- Buzzer actif
- LED
- Résistance 220 Ω
- Arduino Uno

Schéma



Code (extrait)

```
void loop() {
  if (digitalRead(2) == LOW) {
    digitalWrite(8, HIGH); // Buzzer ON
    digitalWrite(9, HIGH); // LED ON
  } else {
    digitalWrite(8, LOW); // Buzzer OFF
    digitalWrite(9, LOW); // LED OFF
  }
}
```

✓ Résultat : Alarme sonore et visuelle activée.
Utilisation : Sécurité, détection d'intrusion.

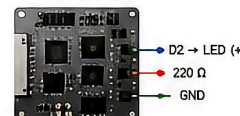
6 PROJET 6 : ESP32 – SERVEUR WEB SIMPLE

L'ESP32 crée un point d'accès WiFi et affiche une page web avec un bouton pour allumer ou éteindre une LED.

Composants

- ESP32
- LED
- Résistance 220 Ω

WiFi & LED



Code (extrait)

```
#include <WiFi.h>

const char* ssid = "DEX_ATOMES";
const char* password = "12345678";

WiFiServer server(80);

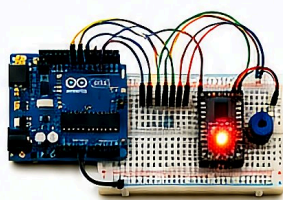
void setup() {
  pinMode(2, OUTPUT);
  WiFi.softAP(ssid, password); server.begin();
}

// Voir le code complet dans les exemples du cours.
```

✓ Résultat : Contrôle d'une LED via navigateur web.
Utilisation : Domotique, IoT, contrôle à distance.

CONSEILS POUR VOS PROJETS

- ✓ Commencez petit, testez souvent.
- ✓ Comprenez chaque composant avant de passer au suivant.
- ✓ Documentez votre code et vos branchements.
- ✓ Utilisez des commentaires dans votre code.
- ✓ Soyez créatif et essayez d'améliorer vos projets !



À RETENIR

- ✓ La pratique est essentielle pour maîtriser Arduino et ESP32.
- ✓ Les projets réels combinent capteurs, actionneurs et logique.
- ✓ Chaque projet peut être adapté à vos besoins.
- ✓ Apprenez en construisant : c'est la clé du succès !
- ✓ Continuez à explorer et à innover.



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

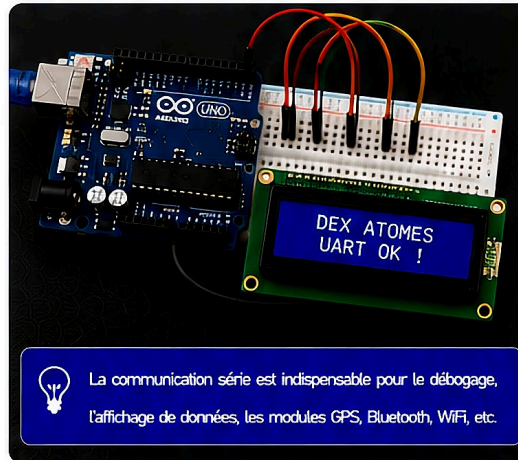
3.16

COMMUNICATION SÉRIE UART ET MONITEUR SÉRIE

Échanger des données entre
l'Arduino / ESP32 et l'ordinateur

La communication série (UART) permet d'envoyer et de recevoir des données texte ou numériques entre votre carte et un ordinateur via USB.

Le Moniteur Série de l'IDE Arduino est un outil puissant pour le débogage et l'affichage des informations en temps réel.



La communication série est indispensable pour le débogage, l'affichage de données, les modules GPS, Bluetooth, WiFi, etc.

TABLE DES MATIÈRES

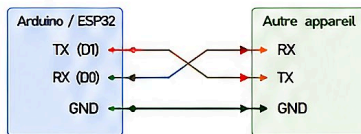
3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74

1 QU'EST-CE QUE L'UART ?

UART (Universal Asynchronous Receiver/Transmitter) est un protocole de communication série asynchrone utilisé pour transmettre des données bit par bit.

Caractéristiques :

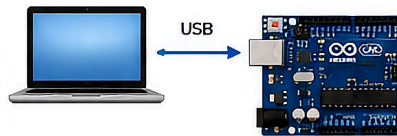
- 2 lignes : TX (Transmission) et RX (Réception)
- Asynchrone : pas d'horloge partagée
- Paramètres : Baudrate, Data bits, Parité, Stop bits
- Utilisé par : GPS, Bluetooth, WiFi, PC, etc.



Connectez toujours GND des deux appareils ensemble pour une communication fiable.

2 CONNEXION AVEC UN ORDINATEUR

La carte est connectée à l'ordinateur via USB.
Le port COM apparaît automatiquement.



Débits (Baudrate) courants :

9600	19200	38400	57600	115200
------	-------	-------	-------	--------

Le Baudrate doit être identique dans le code et dans le Moniteur Série.

3 ENVOYER DES DONNÉES

Utilisez `Serial.print()` ou `Serial.println()` pour envoyer des données vers l'ordinateur.

```
void setup() {
    Serial.begin(9600); // Initialiser la
                        // communication série
    Serial.println("DEX ATOMES - UART Test");
}

void loop() {
    Serial.print("Temp: ");
    Serial.print(25.6);
    Serial.println(" °C");
    delay(1000);
}
```

`println()` ajoute un saut de ligne (`\n`).
Idéal pour lire ligne par ligne.

4 RECEVOIR DES DONNÉES

Utilisez `Serial.available()` pour vérifier si des données sont disponibles, puis `Serial.read()`.

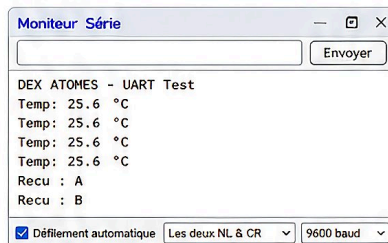
```
void setup() {
    Serial.begin(9600);
    Serial.println("Envoyez quelque chose !");
}

void loop() {
    if (Serial.available() > 0) {
        char c = Serial.read();
        Serial.print("Recu : ");
        Serial.println(c);
    }
}
```

Les données reçues sont un octet (8 bits).
Pour du texte, chaque caractère est envoyé.

5 MONITEUR SÉRIE DE L'IDE ARDUINO

Ouvrez l'IDE → Outils → Moniteur Série (Ctrl+Maj+M) pour afficher les données reçues.



Paramètres importants :

- Baudrate : même valeur que dans le code
- Fin de ligne : Les deux (NL & CR)
- Défilement automatique : activé

Le Moniteur Série est votre allié pour le débogage !

6 EXEMPLE : ÉCHO SÉRIE

Le programme lit un caractère et le renvoie.

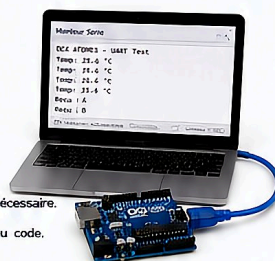
```
void setup() {
    Serial.begin(9600);
    Serial.println("Echo UART - Tapez un
                  caractère");
}

void loop() {
    if (Serial.available() > 0) {
        char c = Serial.read();
        Serial.print("Vous avez envoyé : ");
        Serial.println(c);
    }
}
```

Très utile pour tester des modules (GPS, HC-05, ESP32, etc.) et valider les connexions.

BONNES PRATIQUES

- ✓ Utilisez un Baudrate standard et identique.
- ✓ Vérifiez toujours les connexions TX ↔ RX.
- ✓ Ajoutez des messages clairs pour faciliter le débogage.
- ✓ Ne surchargez pas la communication série (évitez trop d'impressions).
- ✓ Pour de longues données, utilisez des pauses (`delay`) si nécessaire.
- ✓ Fermez le Moniteur Série avant de téléverser un nouveau code.



★ À RETENIR

- ✓ UART permet un échange simple et efficace de données.
- ✓ `Serial.print()` envoie des données vers l'ordinateur.
- ✓ `Serial.available()` et `Serial.read()` reçoivent des données.
- ✓ Le Moniteur Série facilite le débogage et le test en temps réel.
- ✓ La communication série est la base de nombreux projets.
- ✓ Maîtrisez l'UART pour débloquer le plein potentiel de vos projets !



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

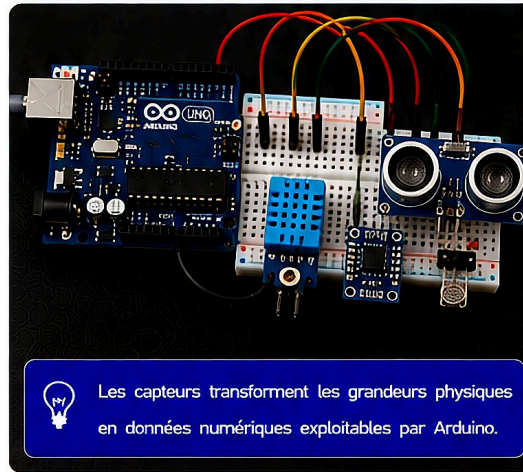
3.17

CAPTEURS ET INTERFACES AVEC ARDUINO

Collecter des données du monde réel avec des capteurs

Les capteurs permettent à votre projet électronique d'interagir avec l'environnement : température, lumière, distance, mouvement, humidité, etc.

Dans cette section, nous découvrons les capteurs les plus courants, comment les connecter et lire leurs valeurs avec Arduino.



Les capteurs transforment les grandeurs physiques en données numériques exploitables par Arduino.

TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	70
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.17	Capteurs et interfaces	75
3.18	Projets avancés	77

1 CAPTEUR DE TEMPÉRATURE (DHT22)

Le DHT22 mesure la température et l'humidité ambiantes.

Connexions

DHT22 → Arduino Uno
VCC → 5V
DATA → D2
GND → GND



Code (extrait)

```
#include <DHT.h>
#define DHTPIN 2
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

void setup(){
  Serial.begin(9600);
  dht.begin();
}

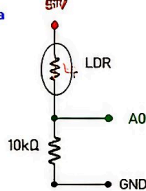
void loop(){
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  Serial.print("T: "); Serial.print(t);
  Serial.print(" C H: "); Serial.print(h);
  Serial.println(" %");
  delay(2000);
}
```

Plage: -40 à 80 °C, 0 à 100 % HR.
Précision élevée et lecture rapide.

2 CAPTEUR DE LUMIÈRE (LDR)

La LDR (photo-résistance) change sa résistance selon la lumière.

Schéma



Code (extrait)

```
int ldrPin = A0;

void setup(){
  Serial.begin(9600);
}

void loop(){
  int val = analogRead(ldrPin);
  Serial.println(val);
  delay(500);
}
```

Plus la lumière est forte, plus la valeur analogique est élevée.

3 CAPTEUR DE DISTANCE (HC-SR04)

Le HC-SR04 utilise des ultrasons pour mesurer la distance.

Connexions

VCC → 5V
TRIG → D9
ECHO → D10
GND → GND



Code (extrait)

```
long duration;
int distance;

void setup(){
  pinMode(9, OUTPUT);
  pinMode(10, INPUT);
  Serial.begin(9600);
}

void loop(){
  digitalWrite(9, LOW); delayMicroseconds(2);
  digitalWrite(9, HIGH); delayMicroseconds(10);
  digitalWrite(9, LOW);
  duration = pulseIn(10, HIGH);
  distance = duration * 0.034 / 2; // cm
  Serial.print("Distance: ");
  Serial.print(distance); Serial.println(" cm");
  delay(500);
}
```

Plage: 2 cm à 400 cm, précision ~3 mm.

4 CAPTEUR DE MOUVEMENT (PIR HC-SR501)

Le PIR détecte la présence d'un mouvement (même de petits animaux).

Connexions

VCC → 5V
OUT → D3
GND → GND



Code (extrait)

```
int pirPin = 3;

void setup(){
  pinMode(pirPin, INPUT);
  Serial.begin(9600);
}

void loop(){
  int val = digitalRead(pirPin);
  if (val == HIGH)
    Serial.println("Mouvement détecté !");
  else
    Serial.println("Aucun mouvement.");
  delay(500);
}
```

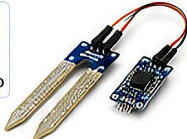
Sortie numérique: HIGH si mouvement détecté, LOW sinon.

5 CAPTEUR D'HUMIDITÉ DU SOL

Mesure le niveau d'humidité du sol (utile pour l'arrosage automatique).

Connexions

VCC → 5V
AOUT → A0
GND → GND



Code (extrait)

```
int soilPin = A0;

void setup(){
  Serial.begin(9600);
}

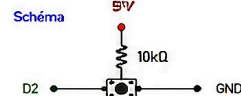
void loop(){
  int val = analogRead(soilPin);
  Serial.print("Humidité: ");
  Serial.println(val);
  delay(1000);
}
```

Valeur faible = sol humide
Valeur élevée = sol sec

6 BOUTON POUSSOIR

Utilisé pour l'entrée utilisateur (on/off, reset, etc.).

Schéma



Code (extrait)

```
const int btnPin = 2;

void setup(){
  pinMode(btnPin, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop(){
  if (digitalRead(btnPin) == LOW) {
    Serial.println("Bouton appuyé");
    delay(300);
  }
}
```

INPUT_PULLUP: pas besoin de résistance externe (le pull-up est interne).

7 CAPTEUR DE GAZ (MQ-2)

Détecte gaz inflammables: LPG, fumée, méthane, alcool, etc.

Connexions

VCC → 5V
GND → GND
AOUT → A0
DOUT → D4 (optionnel)



Code (extrait)

```
int gasPin = A0;

void setup(){
  Serial.begin(9600);
}

void loop(){
  int val = analogRead(gasPin);
  Serial.print("Niveau gaz: ");
  Serial.println(val);
  delay(1000);
}
```

Chauffe interne: attendre 1 à 2 minutes avant des lectures précises.

8 BONNES PRATIQUES



Vérifiez toujours les connexions avant d'alimenter.



Consultez la documentation de chaque capteur.



Utilisez des valeurs de filtrage ou moyenne pour plus de stabilité.



Protégez vos entrées avec des résistances si nécessaire.



Testez vos capteurs avec des exemples simples avant de les intégrer au projet.

★ RÉSUMÉ

- Les capteurs permettent à Arduino de percevoir le monde réel.
- Chaque capteur fournit des données analogiques ou numériques.
- Le câblage correct et la lecture précise sont essentiels.
- Testez et calibrez vos capteurs pour des mesures fiables.



💡 IDÉE PROJET

Créez une station météo connectée : mesurez température, humidité, lumière et affichez les données sur un écran LCD ou envoyez-les via WiFi (ESP32).



3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

3.18

PROJETS AVANCÉS AVEC ARDUINO

Aller plus loin avec des applications concrètes et interactives

Dans cette section, nous réalisons des projets plus complets qui combinent plusieurs composants et concepts : capteurs, affichage, actionneurs et communication.

Ces projets vous permettront de consolider vos compétences et de préparer des systèmes utiles et intelligents.

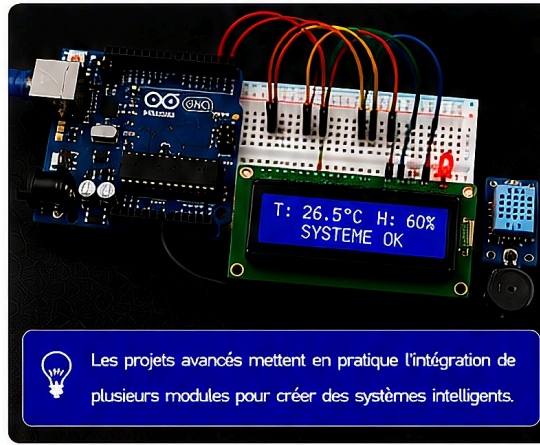


TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Projets avancés	76

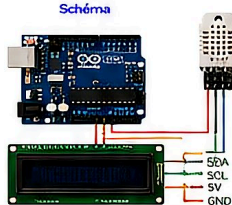
1 STATION MÉTÉO AVEC LCD

Mesure la température et l'humidité puis affiche les données sur un écran LCD 16x2.

Composants

- DHT22
- Écran LCD 16x2 I2C
- Arduino Uno
- Fils de connexion

Schéma



Code (extrait)

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
LiquidCrystal_I2C lcd(0x27,16,2);

void loop(){
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  lcd.setCursor(0,0);
  lcd.print("T:"); lcd.print(t,1); lcd.print(" C");
  lcd.setCursor(0,1);
  lcd.print("H:"); lcd.print(h,0); lcd.print(" %");
  delay(2000);
}
```



Idéal pour surveiller la météo intérieure.

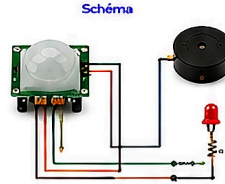
2 SYSTÈME D'ALARME INTELLIGENT

Un capteur PIR détecte un mouvement et déclenche une alarme sonore et une LED clignotante.

Composants

- Capteur PIR HC-SR501
- Buzzer actif
- LED rouge
- Résistance 220 Ω
- Arduino Uno

Schéma



Code (extrait)

```
const int pirPin = 3;
const int buzzerPin = 8;
const int ledPin = 9;

void loop(){
  int etat = digitalRead(pirPin);
  if(etat == HIGH){
    digitalWrite(ledPin, HIGH);
    tone(buzzerPin, 1000);
  }else{
    digitalWrite(ledPin, LOW);
    noTone(buzzerPin);
  }
}
```



Parfait pour sécuriser une pièce ou une entrée.

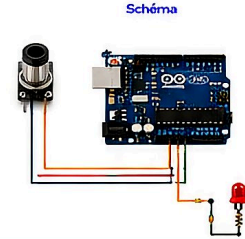
3 CONTRÔLE DE LUMINOSITÉ (PWM)

Utilise un potentiomètre pour ajuster la luminosité d'une LED via PWM.

Composants

- Potentiomètre 10 kΩ
- LED
- Résistance 220 Ω
- Arduino Uno

Schéma



Code (extrait)

```
const int potPin = A0;
const int ledPin = 9;

void loop(){
  int val = analogRead(potPin);
  int pwm = map(val, 0, 1023, 0, 255);
  analogWrite(ledPin, pwm);
}
```



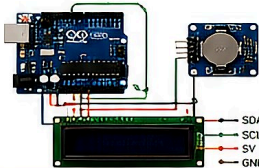
Le PWM permet un contrôle précis de la luminosité.

4 AFFICHAGE D'HEURE AVEC RTC ET LCD

Affiche l'heure réelle en temps réel à l'aide d'un module RTC DS3231.

Composants

- RTC DS3231
- LCD 16x2 I2C
- Arduino Uno
- Fils de connexion



Code (extrait)

```
#include <Wire.h>
#include <RTClib.h>
#include <LiquidCrystal_I2C.h>
RTC_DS3231 rtc;
LiquidCrystal_I2C lcd(0x27,16,2);

void loop(){
  DateTime now = rtc.now();
  lcd.setCursor(0,0);
  lcd.print(now.hour()); lcd.print(":");
  lcd.print(now.minute()); lcd.print(":");
  lcd.print(now.second());
  lcd.setCursor(0,1);
  lcd.print(now.day()); lcd.print("/");
  lcd.print(now.month()); lcd.print("/");
  lcd.print(now.year());
  delay(1000);
}
```



Utilisez le RTC pour des horloges et enregistreurs fiables.

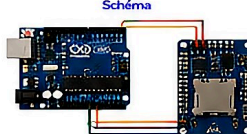
5 DATA LOGGER SUR CARTE SD

Enregistre les données de température (DHT22) sur une carte SD.

Composants

- DHT22
- Module carte SD
- Carte microSD
- Arduino Uno

Schéma



Code (extrait)

```
#include <SPI.h>
#include <SD.h>
#include "DHT.h"
#define DHTPIN 2
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
File dataFile;

void loop(){
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  dataFile = SD.open("data.csv", FILE_WRITE);
  if(dataFile){
    dataFile.print(millis()); dataFile.print(",");
    dataFile.print(t); dataFile.print(",");
    dataFile.print(h); dataFile.print("\n");
    dataFile.close();
  }
  delay(5000);
}
```



Les données peuvent être analysées sur Excel.

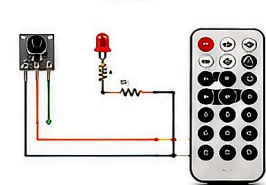
6 COMMANDES À DISTANCE IR

Contrôlez une LED à l'aide d'une télécommande infrarouge.

Composants

- Récepteur IR (VS1838)
- LED
- Résistance 220 Ω
- Télécommande IR
- Arduino Uno

Schéma



Code (extrait)

```
#include <IRremote.h>
const int irPin = 11;
const int ledPin = 9;
IRrecv receiver(irPin);
decode_results results;

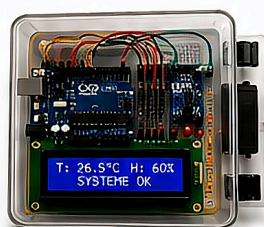
void loop(){
  if(receiver.decode(&results)){
    if(results.value == 0xFA25D0) digitalWrite(ledPin, HIGH); // ON
    if(results.value == 0x9FE21D) digitalWrite(ledPin, LOW); // OFF
    receiver.resume();
  }
}
```



Les codes IR peuvent varier selon la télécommande.

BONNES PRATIQUES

- ✓ Testez chaque module séparément avant l'intégration finale.
- ✓ Commentez toujours votre code.
- ✓ Utilisez des noms de variables clairs et significatifs.
- ✓ Protégez vos entrées avec des résistances si nécessaire.
- ✓ Sauvegardez régulièrement vos données (SD ou EEPROM).
- ✓ Protégez vos montages contre les inversions de polarité.



À RETENIR

- ✓ Les projets avancés combinent plusieurs composants.
- ✓ Les capteurs collectent des données du monde réel.
- ✓ Les affichages et actionneurs permettent une interaction.
- ✓ La programmation rend le système intelligent et utile.
- ✓ Chaque projet peut être amélioré et personnalisé.
- ✓ La créativité est la clé de l'innovation en électronique !



3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

3.19

INTERRUPTIONS EXTERNES ET BOUTONS AVANCÉS AVEC ARDUINO

Réagir instantanément à un événement extérieur avec les interruptions.

Les interruptions permettent à Arduino de répondre immédiatement à un signal externe sans attendre la fin du code en cours.

Elles sont idéales pour les boutons, les capteurs rapides, les encodeurs et toutes les situations où le timing est critique.

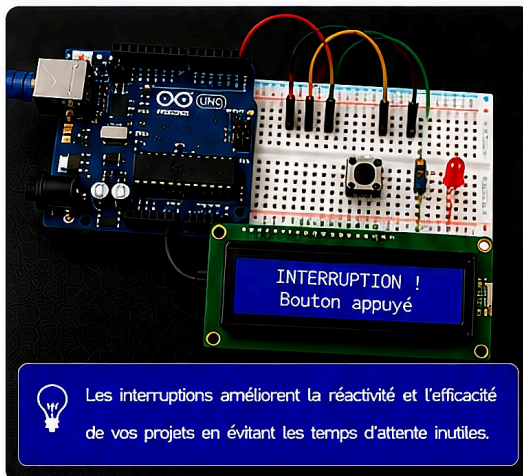
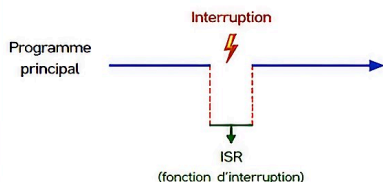


TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Projets avancés	76
3.19	Interruptions externes	77
3.20	Projets avancés avec interruptions	78

1 QU'EST-CE QU'UNE INTERRUPTION ?

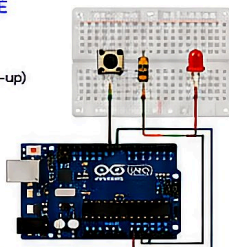
Une interruption est un signal qui interrompt le programme principal pour exécuter une fonction spéciale (ISR).



Arduino peut gérer plusieurs interruptions configurées sur des broches spécifiques.

2 MATÉRIEL UTILISÉ

- Bouton poussoir
- Résistance 10 kΩ (pull-up)
- LED
- Arduino Uno
- Fils de connexion



Connexion (exemple avec D2)

Bouton	→ D2 (avec pull-up)
LED	→ D13
10 kΩ	→ entre D2 et 5V
GND	→ commun

La résistance pull-up maintient la broche à HIGH quand le bouton n'est pas appuyé.

3 INTERRUPTION EXTERNE SUR D2

attachInterrupt() permet d'exécuter une fonction automatiquement quand un événement se produit.

```
volatile bool boutonAppuye = false; // variable

void ISR_bouton() {
    boutonAppuye = true; // ISR très courte
}

void setup() {
    pinMode(2, INPUT_PULLUP);
    pinMode(13, OUTPUT);
    attachInterrupt(digitalPinToInterrupt(2),
        ISR_bouton, FALLING); // front descendant
}
```

Les ISR doivent être courtes et rapides. Évitez les delays et Serial.print() dans l'ISR.

4 CODE PRINCIPAL (LOOP)

Le programme principal vérifie la variable mise à jour par l'ISR.

```
void loop() {
    if (boutonAppuye) {
        digitalWrite(13, !digitalRead(13)); // toggle LED
        boutonAppuye = false; // réinitialiser
    }
}
```

Nous utilisons une variable volatile pour partager l'information entre l'ISR et le programme principal.

5 ANTI-REBONDISSEMENT (DEBOUNCE)

Les boutons mécaniques produisent plusieurs signaux rapides. Utilisons un temps minimal.

```
volatile bool boutonAppuye = false;
unsigned long dernierTemps = 0;
const unsigned long delaiDebounce = 50; // ms

void ISR_bouton() {
    unsigned long maintenant = millis();
    if (maintenant - dernierTemps > delaiDebounce) {
        boutonAppuye = true;
        dernierTemps = maintenant;
    }
}
```

Le debounce garantit une détection fiable d'un seul appui.

6 CHANGER LE MODE D'INTERRUPTION

Vous pouvez choisir le type d'interruption selon votre besoin.

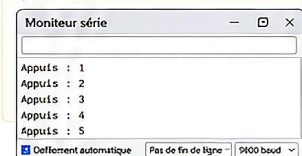
Mode	Déclenchement	Utilisation typique
LOW	Niveau bas	Signal maintenu bas
CHANGE	Changement d'état	Encodeurs, signaux AC
RISING	Front montant	Appuyer (relâchement)
FALLING	Front descendant	Appuyer (enfoncement)

Utilisez attachInterrupt(pin, ISR, mode) pour spécifier le mode.

★ EXEMPLE COMPLET : BOUTON INTERRUPTION AVEC LED ET COMPTEUR

```
volatile bool boutonAppuye = false;
volatile int compteur = 0;
unsigned long dernierTemps = 0;
const unsigned long delaiDebounce = 50;
```

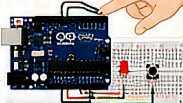
```
void ISR_bouton() {
    unsigned long t = millis();
    if (t - dernierTemps > delaiDebounce) {
        boutonAppuye = true;
        dernierTemps = t;
    }
}
```



Chaque appui incrémente le compteur et change l'état de la LED. Vérifiez les messages dans le Moniteur série.

```
void setup() {
    pinMode(2, INPUT_PULLUP);
    pinMode(13, OUTPUT);
    Serial.begin(9600);
    attachInterrupt(digitalPinToInterrupt(2),
        ISR_bouton, FALLING);
}

void loop() {
    if (boutonAppuye) {
        compteur++;
        digitalWrite(13, !digitalRead(13));
        Serial.print("Appuis : ");
        Serial.println(compteur);
        boutonAppuye = false;
    }
}
```



★ BONNES PRATIQUES

- ✓ Définissez correctement le mode d'interruption.
- ✓ Gardez les ISR courtes et simples.
- ✓ Utilisez des variables volatile pour partager des données.
- ✓ Protégez contre les rebonds des boutons.
- ✓ Testez toujours vos interruptions avec le Moniteur Série.
- ✓ Documentez vos broches d'interruption utilisées.



i INFO

Sur Arduino Uno : Interruptions externes disponibles sur D2 et D3.

Sur ESP32 : beaucoup plus de broches avec interruptions (voir documentation ESP32).



3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

3.20

AFFICHEURS LCD ET 7 SEGMENTS AVEC ARDUINO

Afficher des informations claires avec des écrans LCD et des afficheurs 7 segments.

Les afficheurs permettent de communiquer des données à l'utilisateur : texte, nombres, messages ou états du système.

Dans cette section, nous apprenons à utiliser un écran LCD 16x2 avec I2C et un afficheur 7 segments à 4 chiffres.

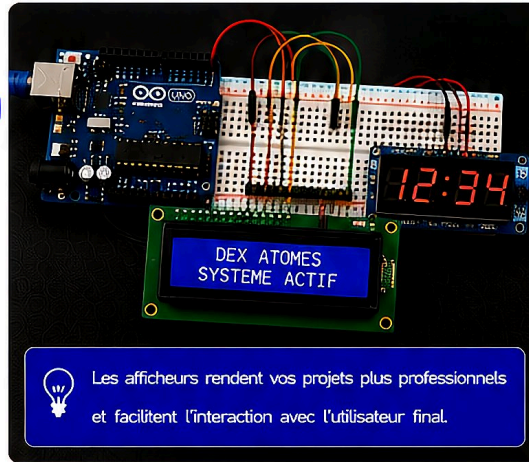


TABLE DES MATIÈRES

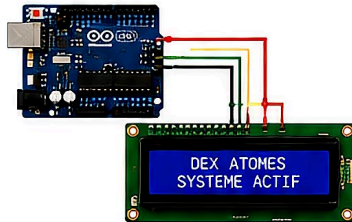
3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques ...	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées ..	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Projets avancés	76
3.19	Interruptions externes	77
3.20	Afficheurs LCD et 7 segments	78
3.21	Matrices LED et animations	79

1 ÉCRAN LCD 16x2 AVEC MODULE I2C

L'interface I2C permet de réduire le câblage à seulement 4 fils.

Connexions

LCD I2C	→	Arduino Uno
VCC	→	5V
GND	→	GND
SDA	→	A4 (SDA)
SCL	→	A5 (SCL)



Code (extrait)

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2); // Adresse I2C (peut varier)

void setup() {
  lcd.init();           // Initialiser l'écran
  lcd.backlight();      // Allumer le rétroéclairage
  lcd.setCursor(0, 0);  // Ligne 1, Colonne 0
  lcd.print("DEX ATOMES");
  lcd.setCursor(0, 1);  // Ligne 2, Colonne 0
  lcd.print("SYSTEME ACTIF");
}

void loop() {
  lcd.setCursor(0, 1);
  lcd.print("T: 25.6C H: 60%");
  delay(1000);
}
```

Note :

L'adresse I2C la plus courante est 0x27 ou 0x3F. Utilisez un sketch de scan I2C si l'écran n'affiche rien.

Résultat

Affichage de texte, messages, mesures et états du système sur un écran clair et lisible.

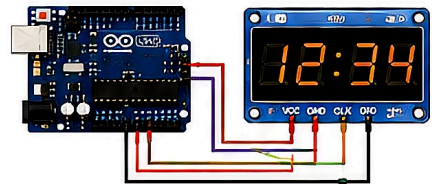


2 AFFICHEUR 7 SEGMENTS 4 CHIFFRES (TM1637)

Le module TM1637 simplifie le câblage et le contrôle de l'afficheur.

Connexions

TM1637	→	Arduino Uno
VCC	→	5V
GND	→	GND
CLK	→	D2
DIO	→	D3



Code (extrait)

```
#include <TM1637Display.h>
#define CLK 2
#define DIO 3

TM1637Display display(CLK, DIO);

void setup() {
  display.setBrightness(7); // 0 à 7
}

void loop() {
  display.showNumberDec(1234, true); // Affiche 1234
  delay(1000);
  display.showNumberDecEx(5678, 0b01000000, true); // Affiche 56:78
  delay(1000);
}
```

Astuce :

Vous pouvez afficher l'heure, des compteurs, des températures ou des valeurs numériques.

Résultat

Affichage rapide et efficace de nombres ou d'horloges sur 4 chiffres.



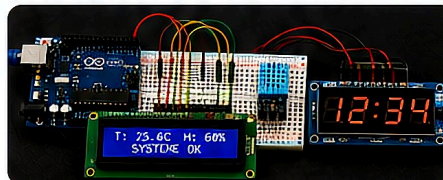
BONNES PRATIQUES

- ✓ Vérifiez les connexions avant de télécharger le code.
- ✓ Utilisez des résistances de limitation pour protéger les LEDs des afficheurs.
- ✓ Réglez la luminosité selon l'environnement.
- ✓ Gardez vos fils courts et bien organisés.
- ✓ Utilisez des modèles de modules fiables (I2C, TM1637).

EXEMPLE DE PROJET

Station météo avec LCD et horloge :

- Affiche la température et l'humidité sur le LCD.
- Affiche l'heure en temps réel sur le 7 segments.



À RETENIR

- ✓ L'écran LCD I2C simplifie l'affichage de textes.
- ✓ Le TM1637 facilite l'utilisation des afficheurs 7 segments.
- ✓ Les afficheurs améliorent l'interface et la convivialité de vos projets.
- ✓ Combinez les afficheurs avec des capteurs pour des systèmes intelligents.

DÉPANNAGE RAPIDE

Problème	Cause possible	Solution
LCD n'affiche rien	Mauvaise adresse I2C	Essayez 0x27 ou 0x3F
LCD affiche des carrés	Contraste trop faible	Ajustez le potentiomètre du module
7 segments éteint	Mauvaise connexion VCC/GND	Vérifiez l'alimentation et le câblage
Chiffres instables	Fils trop longs / parasites	Raccourcissez les fils et vérifiez GND

RESSOURCES UTILES

Bibliothèque LiquidCrystal_I2C : github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library

Bibliothèque TM1637Display : github.com/avishorp/TM1637

Scanner I2C : recherchez "I2C Scanner Arduino"



Scannez pour plus de ressources !

3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

3.21

MATRICES LED ET ANIMATIONS AVEC ARDUINO

Créer des affichages dynamiques et des animations à l'aide d'une matrice LED 8x8.

Les matrices LED permettent d'afficher des images, icônes, texte défilant et animations personnalisées. Elles sont idéales pour les projets interactifs et les systèmes d'information visuelle.

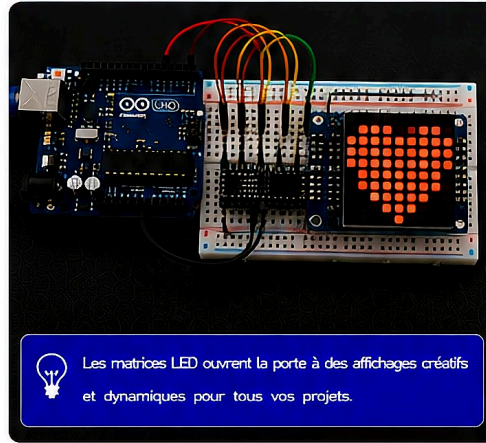


TABLE DES MATIÈRES

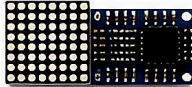
3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Projets avancés	76
3.19	Interruptions externes	77
3.20	Projets avancés avec interruptions	78
3.21	Matrices LED et animations	79

1 PRÉSENTATION DU MATÉRIEL

Nous utilisons une matrice LED 8x8 avec le driver MAX7219 pour simplifier le câblage.

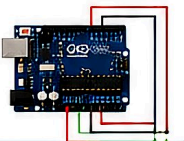
Composants

- Matrice LED 8x8
- Module MAX7219
- Arduino Uno
- Fils de connexion



Connexions (MAX7219 ↔ Arduino Uno)

VCC → 5V
GND → GND
DIN → D11 (MOSI)
CLK → D13 (SCK)
CS → D10 (SS)



Le MAX7219 contrôle la matrice avec seulement 3 broches numériques de l'Arduino.

2 INSTALLER LA BIBLIOTHÈQUE

Nous utilisons la bibliothèque LedControl pour piloter facilement la matrice.

Étapes

1. Ouvrez l'IDE Arduino.
2. Allez dans Croquis → Inclure une bibliothèque → Gérer les bibliothèques...
3. Recherchez LedControl.
4. Installez la bibliothèque de Wayoda.

Cette bibliothèque simplifie l'envoi de données à la matrice via le MAX7219.

3 INITIALISATION DU CODE

Incluez la bibliothèque et créez l'objet LedControl.

```
#include <LedControl.h>

// Broches de connexion
int DIN = 11; // MOSI
int CLK = 13; // SCK
int CS = 10; // SS

int nombreMatrices = 1; // Nombre de matrices

LedControl lc = LedControl(DIN, CLK, CS,
    nombreMatrices);

void setup() {
    lc.shutdown(0, false); // Sortie de veille
    lc.setIntensity(0, 8); // Luminosité (0-15)
    lc.clearDisplay(0); // Efface l'écran
}
```

La luminosité est réglable de 0 (éteint) à 15 (max).

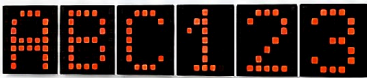
4 AFFICHER UN CARACTÈRE

Affiche un caractère (A-Z, 0-9) sur la matrice.

```
void loop() {
    char lettre = 'A';
    afficherCaractere(lettre);
    delay(2000);
}

void afficherCaractere(char c) {
    lc.clearDisplay(0);
    lc.setChar(0, 3, c, false);
}
```

Exemples de caractères



Les caractères sont centrés automatiquement. La colonne de départ peut être ajustée.

5 ANIMATION DÉFILANTE

Faites défiler un texte sur la matrice LED.

```
String message = "DEX ATOMES";
int scrollPos = 0;

void loop() {
    lc.clearDisplay(0);
    for (int i = 0; i < message.length(); i++) {
        scrollText(message, scrollPos + i);
    }
    scrollPos++;
    if (scrollPos > message.length() * 8) scrollPos = 0;
    delay(150);
}
```

Effet obtenu



Le texte défile de la droite vers la gauche. La vitesse dépend du delay().

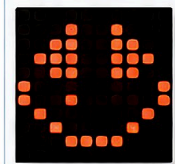
6 ANIMATION PERSONNALISÉE

Affichez des motifs, icônes ou animations.

```
byte smile[8] = {
    000111100,
    001000010,
    010100101,
    010000001,
    010011001,
    010100101,
    001000010,
    000111100
};

void loop() {
    lc.clearDisplay(0);
    for (int i = 0; i < 8; i++) {
        lc.setRow(0, i, smile[i]);
    }
    delay(2000);
}
```

Icône affichée



Créez vos propres icônes avec un éditeur binaire ou directement dans le code.

BONNES PRATIQUES

- ✓ Réglez la luminosité pour éviter d'éblouir.
- ✓ Utilisez des messages courts pour un meilleur défilement.
- ✓ Évitez les delays trop longs dans les animations.
- ✓ Séparez la logique d'affichage du reste du code.
- ✓ Testez vos animations étape par étape.

EXEMPLE COMPLET

Un exemple complet combinant texte fixe et texte défilant est disponible dans les ressources (voir QR code).



Scanner pour voir le code

À RETENIR

- ✓ La matrice LED 8x8 avec MAX7219 est facile à interfacer.
- ✓ La bibliothèque LedControl simplifie le développement.
- ✓ Vous pouvez afficher du texte, des nombres et des icônes.
- ✓ Les animations rendent vos projets plus attractifs et professionnels.

3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

3.22

ACTIONNEURS ET SORTIES NUMÉRIQUES AVEC ARDUINO

Piloter le monde réel : LED, relais, moteurs, servos et buzzers.

Les actionneurs permettent à votre projet d'interagir avec l'environnement en produisant du mouvement, de la lumière, du son ou d'autres actions physiques.

Dans cette section, nous apprenons à contrôler différents actionneurs à l'aide des sorties numériques et PWM d'Arduino.

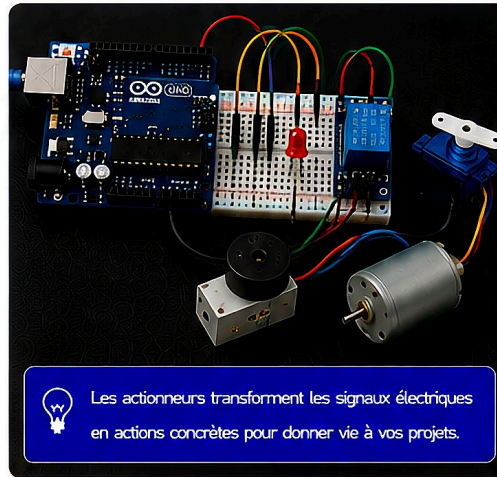


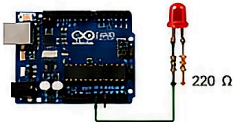
TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	80
3.11	Afficheurs LCD	81
3.12	Matrices LED et 7 segments	82
3.13	Stockage et cartes mémoire	83
3.14	Horloge temps réel (RTC)	84
3.15	Projets pratiques	85
3.16	Communication série UART	86
3.17	Capteurs et interfaces	87
3.18	Projets avancés	88
3.19	Interruptions externes	89
3.20	Projets avancés avec interruptions	90
3.21	Matrices LED et animations	91
3.22	Actionneurs et sorties numériques	80

1 ALLUMER UNE LED

Le moyen le plus simple pour contrôler une charge : une LED.

Schéma



Connexions

LED + → D13
LED - → GND
Résistance
220 Ω en série

Code (extrait)

```
void setup() {
  pinMode(13, OUTPUT);
}

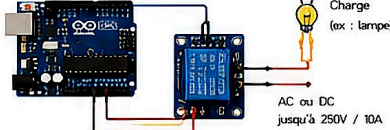
void loop() {
  digitalWrite(13, HIGH); // LED ON
  delay(1000);
  digitalWrite(13, LOW); // LED OFF
  delay(1000);
}
```

La broche D13 est souvent utilisée pour la LED intégrée.

2 PILOTER UN RELAIS

Le relais permet de contrôler des appareils fonctionnant sous une tension plus élevée.

Schéma



Code (extrait)

```
void setup() {
  pinMode(8, OUTPUT);
}

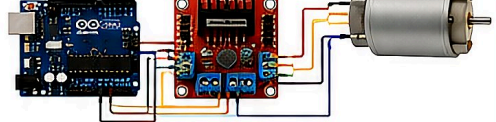
void loop() {
  digitalWrite(8, HIGH); // Relais activé
  delay(2000);
  digitalWrite(8, LOW); // Relais désactivé
  delay(2000);
}
```

Utilisez un relais pour isoler et protéger votre Arduino.

3 CONTRÔLER UN MOTEUR DC

Un moteur DC peut être contrôlé avec un transistor ou un module L298N.

Schéma



Connexions (L298N)

ENA → D5 (PWM)
IN1 → D7
IN2 → D8
VCC → 5V
GND → GND
+12V → Alim moteur

Code (extrait)

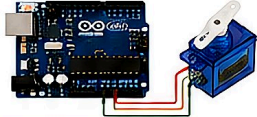
```
void loop() {
  digitalWrite(7, HIGH);
  digitalWrite(8, LOW); // Sens 1
  analogWrite(5, 180); // Vitesse (0-255)
  delay(3000);
  digitalWrite(7, LOW);
  digitalWrite(8, HIGH); // Sens 2
  delay(3000);
}
```

Le PWM sur ENA permet de contrôler la vitesse.

4 UTILISER UN SERVO MOTEUR

Un servo moteur permet un contrôle précis de position (0° à 180°).

Schéma



Connexions

Signal → D9
VCC → 5V
GND → GND

Code (extrait)

```
#include <Servo.h>
Servo monServo;

void setup() {
  monServo.attach(9);
}

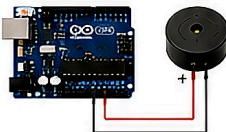
void loop() {
  for (int pos = 0; pos <= 180; pos += 1) {
    monServo.write(pos);
    delay(15);
  }
  for (int pos = 180; pos >= 0; pos -= 1) {
    monServo.write(pos);
    delay(15);
  }
}
```

Alimentez le servo avec une source stable de 5V.

5 FAIRE SONNER UN BUZZER

Le buzzer piézo émet un son lorsque la broche est à HIGH.

Schéma



Connexions

+ → D10
- → GND

Code (extrait)

```
void loop() {
  digitalWrite(10, HIGH); // Son ON
  delay(1000);
  digitalWrite(10, LOW); // Son OFF
  delay(1000);
}
```

Certains buzzers actifs ont un oscillateur intégré.

6 COMMANDER PLUSIEURS SORTIES

Combinez plusieurs actionneurs pour des projets interactifs et intelligents.

Exemple de schéma



Idées de projets :

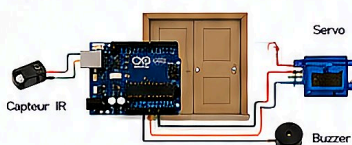
- Système d'alarme
- Porte automatique
- Robot suiveur de ligne
- Maquette domotique
- Bras robotisé
- Voiture télécommandée

Planifiez votre câblage et utilisez des alimentations adaptées à chaque composant.

★ BONNES PRATIQUES

- ✓ Utilisez toujours une résistance pour les LED.
- ✓ Ne dépassez pas le courant maximum des broches (20 mA recommandé).
- ✓ Utilisez des alimentations séparées pour les moteurs et gros modules.
- ✓ Protégez vos circuits avec des diodes de roue libre pour les charges inductives (relais, moteurs).
- ✓ Testez chaque actionneur séparément avant de tout intégrer dans votre projet.

🔧 EXEMPLE DE PROJET : PORTE AUTOMATIQUE



Le capteur détecte une personne, la porte s'ouvre avec le servo et un buzzer confirme l'action.

🎓 À RETENIR

- ✓ Les actionneurs permettent d'agir sur le monde réel.
- ✓ Les sorties numériques (HIGH/LOW) contrôlent l'état ON/OFF.
- ✓ Le PWM permet de varier la vitesse ou la puissance.
- ✓ Les modules (relais, drivers, servos) simplifient le câblage et protègent l'Arduino.
- ✓ Un bon projet combine capteurs + actionneurs + logique !

3

3.23

CAPTEURS ET INTERFACES AVEC ARDUINO

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

Lire le monde réel et interagir avec vos projets.

Les capteurs permettent de mesurer des grandeurs physiques comme la température, la distance, la lumière, etc.

Dans cette section, nous découvrons les capteurs les plus utilisés et les interfaces pour communiquer des données.

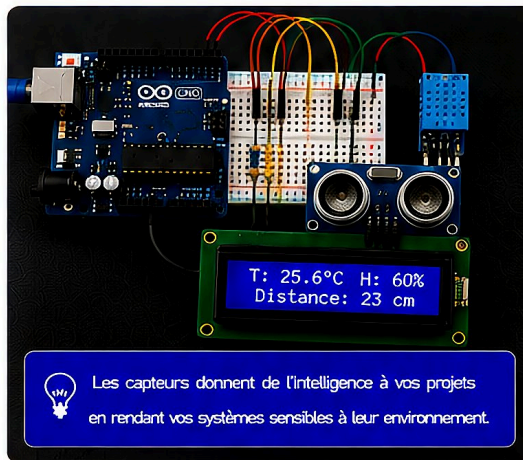


TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
	Communication I2C	
3.9	Capteurs et interface	81
3.10	Actuateurs et sorties avancées	82
3.11	Afficheurs LCD	82
3.11	Afficheurs LCD	83
3.12	Matrices LED et 7 segments	84
3.13	Stockage et cartes mémoire	85
3.14	Horloge temps réel (RTC)	86
3.15	Projets pratiques	87
3.16	Communication série UART	88
3.17	Capteurs et interfaces	89
3.18	Interruptions externes	90
3.19	Projets avancés avec interruptions	90
3.20	Matrices LED et animations	91
3.23	Capteurs et interfaces	81

CAPTEURS POPULAIRES

1 DHT22 – TEMPÉRATURE ET HUMIDITÉ



Caractéristiques

- Température : -40 à 80°C
- Humidité : 0 à 100%
- Précision élevée

Branchement

VCC → 5V
DATA → D2
GND → GND

Code (extrait)

```
#include <DHT.h>
#define DHTPIN 2
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

void setup(){
  Serial.begin(9600);
  dht.begin();
}

void loop(){
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  Serial.print("T: "); Serial.print(t);
  Serial.print("°C H: "); Serial.print(h);
  Serial.println(" %");
  delay(2000);
}
```



Idéal pour les stations météo et les systèmes HVAC.

2 HC-SR04 – CAPTEUR À ULTRASONS



Caractéristiques

- Mesure : 2 à 400 cm
- Précision : ±3 mm
- Angle : 15°

Code (extrait)

```
long duration;
int distance;

void setup(){
  pinMode(9, OUTPUT);
  pinMode(10, INPUT);
  Serial.begin(9600);
}

void loop(){
  digitalWrite(9, LOW); delayMicroseconds(2);
  digitalWrite(9, HIGH); delayMicroseconds(10);
  digitalWrite(9, LOW);
  duration = pulseIn(10, HIGH);
  distance = duration * 0.034 / 2;
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
  delay(500);
}
```



Parfait pour mesurer des distances et éviter les obstacles.

3 LDR – CAPTEUR DE LUMINOSITÉ



Caractéristiques

- Réagit à la lumière
- Sortie analogique
- Simple et économique

Branchement

LDR → A0
Résistance 10kΩ → GND
5V → +

Code (extrait)

```
int ldrPin = A0;

void setup(){
  Serial.begin(9600);
}

void loop(){
  int valeur = analogRead(ldrPin);
  int pourcentage = map(valeur, 0, 1023, 0, 100);
  Serial.print("Luminosite: ");
  Serial.print(pourcentage);
  Serial.println(" %");
  delay(500);
}
```



Utiliser dans les systèmes d'éclairage automatique et les panneaux solaires.

4 JOYSTICK – MODULE À 2 AXES



Caractéristiques

- 2 axes : X et Y
- Bouton intégré (SW)
- Sorties analogiques

Branchement

VCC → 5V
VRx → A0
VRY → A1
SW → D3
GND → GND

Code (extrait)

```
int ldrPin = A0;
int yPin = A1;
int swPin = 3;

void setup(){
  pinMode(swPin, INPUT_PULLUP);
  Serial.begin(9600);
}

void loop(){
  int x = analogRead(xPin);
  int y = analogRead(yPin);
  int bouton = digitalRead(swPin);
  Serial.print("X:"); Serial.print(x);
  Serial.print(" Y:"); Serial.print(y);
  Serial.print(" SW:"); Serial.println(bouton);
  delay(200);
}
```



Idéal pour les robots, drones et interfaces de contrôle.

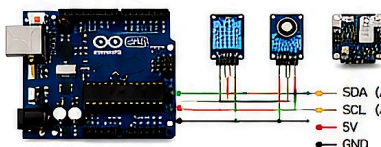
INTERFACES DE COMMUNICATION

I2C (Inter-Integrated Circuit)

Interface série à 2 fils (SDA, SCL) pour connecter plusieurs périphériques avec peu de câbles.

Broches Arduino Uno :

- SDA → A4
- SCL → A5

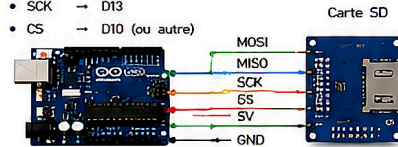


SPI (Serial Peripheral Interface)

Interface rapide à 4 fils (plus CS) pour les périphériques nécessitant un haut débit.

Broches Arduino Uno (matériel SPI) :

- MOSI → D11
- MISO → D12
- SCK → D13
- CS → D10 (ou autre)

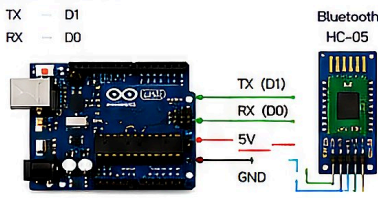


UART (Série)

Communication série asynchrone pour échanger des données simplement entre deux appareils.

Broches Arduino Uno :

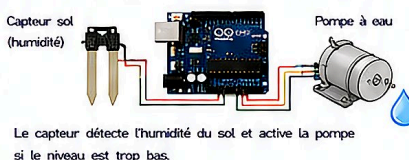
- TX → D1
- RX → D0



BONNES PRATIQUES

- ✓ Calibrez vos capteurs pour des mesures précises.
- ✓ Filtrez les données bruitées avec la moyenne mobile.
- ✓ Utilisez des résistances de tirage (pull-up/pull-down) si nécessaire.
- ✓ Gardez vos câbles courts et bien organisés.
- ✓ Testez chaque capteur indépendamment avant l'intégration finale.

EXEMPLE DE PROJET : SYSTÈME D'ARRROSAGE AUTOMATIQUE



À RETENIR

- ✓ Les capteurs permettent à Arduino de percevoir le monde réel.
- ✓ Choisissez le bon capteur pour chaque mesure.
- ✓ I2C, SPI et UART sont des interfaces essentielles pour communiquer.
- ✓ Un bon câblage et un code propre garantissent des mesures fiables.

3

CHAPITRE 3 Arduino et ESP32 – Fondamentaux

324

COMMUNICATION SÉRIE ET VISUALISATION DES DONNÉES

Envoyer et recevoir des données entre Arduino et un ordinateur ou un autre appareil.

La communication série est essentielle pour le débogage, l'affichage de mesures en temps réel et l'échange de commandes.

Nous verrons Serial Monitor, plotter et communication avec d'autres plateformes.

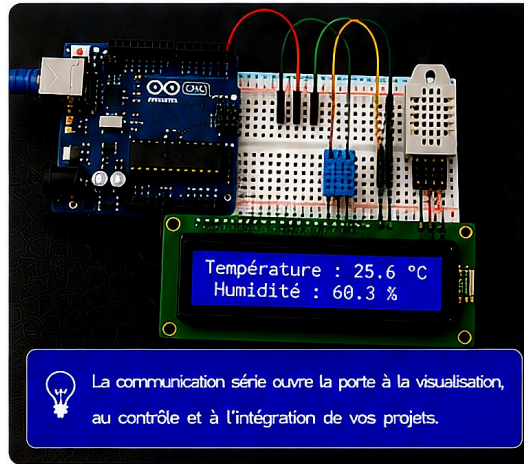


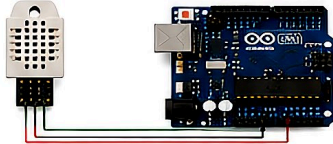
TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Interruptions externes	76
3.19	Projets avancés avec interruptions	77
3.20	Matrices LED et animations	79
3.21	Actionneurs et sorties numériques	80
3.22	Capteurs et interfaces	81

3.24 Communication série et visualisation.82

1 ENVOYER DES DONNÉES AU MONITEUR SÉRIE

Utilisons un capteur de température (DHT22) pour afficher les valeurs en temps réel.



Code (extrait)

```
#include <DHT.h>
#define DHTPIN 2
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

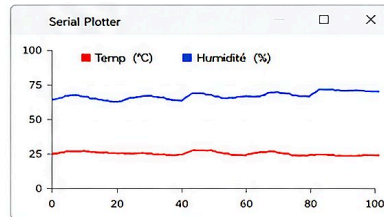
void setup() {
  Serial.begin(9600); // Démarre la communication série
  dht.begin();
}

void loop() {
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  if (!isnan(t) || !isnan(h)) return;
  Serial.print("T: "); Serial.print(t);
  Serial.print(" °C H: "); Serial.print(h);
  Serial.println(" %");
  delay(2000);
}
```

Ouvrez l'outil "Moniteur série" à 9600 bauds.

2 AFFICHER SUR LE PLOTTER SÉRIE

Le Serial Plotter permet de visualiser l'évolution des mesures graphiquement.



Code (extrait)

```
void loop() {
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  if (!isnan(t) || !isnan(h)) return;
  Serial.print(t); // 1ère courbe (Temp)
  Serial.print(" ");
  Serial.print(h); // 2ème courbe (Humidité)
  delay(1000);
}
```

Chaque valeur doit être séparée par un espace et suivie d'un retour à la ligne.

3 RECEVOIR DES COMMANDES

Arduino peut recevoir des commandes depuis l'ordinateur via le Moniteur série.

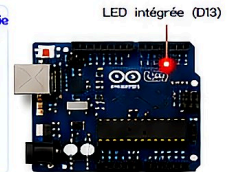
Code (extrait)

```
void setup() {
  Serial.begin(9600);
  pinMode(13, OUTPUT); // LED intégrée
}

void loop() {
  if (Serial.available() > 0) {
    char cmd = Serial.read();
    if (cmd == '1') digitalWrite(13, HIGH);
    if (cmd == '0') digitalWrite(13, LOW);
    Serial.print("Commande reçue: ");
    Serial.println(cmd);
  }
}
```

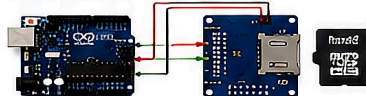
Envoyez dans le Moniteur série

- 1 → Allume la LED intégrée
- 0 → Éteint la LED intégrée



4 ENREGISTRER LES DONNÉES SUR CARTE SD

Enregistrez les mesures dans un fichier texte pour analyse ultérieure.



Code (extrait)

```
#include <SPI.h>
#include <SD.h>
File dataFile; // Fichier sur la carte SD

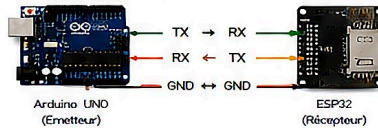
void setup() {
  Serial.begin(9600);
  SD.begin(10); // Broche CS de la carte SD
  dataFile = SD.open("data.txt", FILE_WRITE);
}

void loop() {
  float t = dht.readTemperature();
  dataFile.print(millis()); dataFile.print(",");
  dataFile.println(t);
  delay(2000);
}
```

Idéal pour l'enregistrement à long terme et l'analyse dans Excel ou Python.

5 ENVOYER DES DONNÉES VERS UN ESP32

Communication série entre deux cartes via TX/RX.



Code (extrait) – Arduino (émetteur)

```
void setup() { Serial.begin(9600); }
void loop() {
  Serial.println("Hello ESP32");
  delay(1000);
}
```

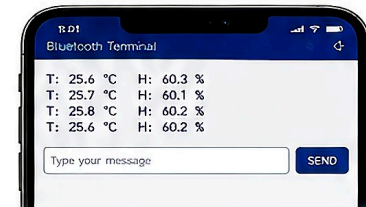
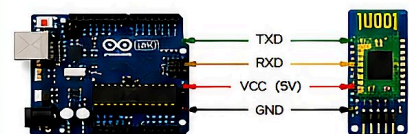
Code (extrait) – ESP32 (récepteur)

```
void setup() { Serial.begin(9600); }
void loop() {
  if (Serial.available()) {
    String msg = Serial.readStringUntil('\n');
    Serial.println("Reçu: " + msg);
  }
}
```

Vérifiez que les vitesses (baud rate) sont identiques sur les deux cartes.

6 COMMUNICATION BLUETOOTH (HC-05)

Envoyez des données vers votre smartphone.

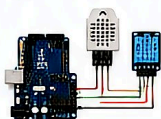


Parfait pour les applications mobiles et le contrôle à distance de vos projets.

BONNES PRATIQUES

- Utilisez des vitesses standards : 9600, 19200, 115200.
- Toujours partager la masse (GND) entre les appareils.
- Ajoutez des messages clairs pour faciliter le débogage.
- Ne surchargez pas le port série, ajoutez des détails si besoin.
- Sauvegardez vos données pour analyse future.

EXEMPLE COMPLET : STATION MÉTÉO



- Mesure T° et Humidité (DHT22)
- Affiche sur l'écran LCD
- Enregistre sur carte SD
- Envoie vers smartphone (Bluetooth)

Un projet complet intégrant capteur, affichage, stockage et communication sans fil !

À RETENIR

- La communication série est un outil puissant pour visualiser et contrôler vos projets.
- Le Serial Plotter permet de voir les données sous forme de graphique en temps réel.
- Vous pouvez échanger des données avec d'autres cartes, un ordinateur ou un smartphone.
- Exploitez ces techniques pour créer des systèmes intelligents et connectés !

3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

3.25

PWM AVANCÉ ET CONTRÔLE DES MOTEURS

Utiliser la modulation de largeur d'impulsion (PWM) pour contrôler précisément la vitesse et la puissance des moteurs.

Le PWM permet de simuler une tension analogique en envoyant des impulsions numériques à haute fréquence.

Nous approfondirons la génération du PWM et son application pour piloter des moteurs DC avec plus d'efficacité.

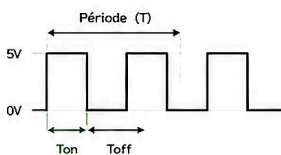


TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Projets avancés	76
3.19	Interruptions externes	77
3.20	Projets avancés avec interruptions	78
3.21	Matrices LED et animations	79
3.22	Actionneurs et sorties numériques	80
3.23	Capteurs et interfaces	81
3.24	Communication série et visualisation	82
3.25	PWM avancé et contrôle des moteurs	83

1 QU'EST-CE QUE LE PWM ?

Le PWM (Pulse Width Modulation) consiste à envoyer des impulsions à haute fréquence avec un rapport cyclique variable.

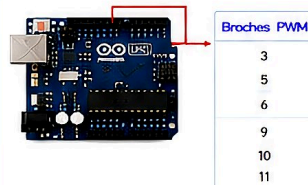


$$\text{Rapport cyclique (\%)} = \frac{T_{\text{on}}}{T} \times 100$$

Plus le Ton est long, plus la puissance moyenne est élevée.

2 GÉNÉRER DU PWM AVEC ARDUINO

Les broches marquées ~ sont capables de générer du PWM.



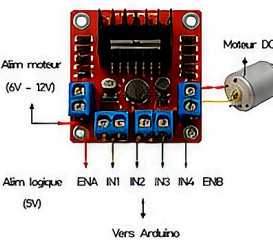
```
void setup() {
  pinMode(9, OUTPUT); // Broche PWM
}

void loop() {
  analogWrite(9, 128); // 0 à 255
  delay(1000);
}
```

analogWrite() : 0 = 0% (OFF) à 255 = 100% (ON)

3 CONTRÔLER UN MOTEUR DC AVEC L298N

Le module L298N permet de contrôler la direction et la vitesse d'un moteur DC.



Connexions (exemple)

- ENA → 9 (PWM vitesse)
- IN1 → 8 (Direction 1)
- IN2 → 7 (Direction 2)
- IN3 → 5 (Non utilisé ici)
- IN4 → 6 (Non utilisé ici)

ENA contrôle la vitesse, IN1 et IN2 la direction.

4 EXEMPLE : VARIER LA VITESSE

Ce code augmente et diminue la vitesse du moteur progressivement.

```
Code (extrait)

const int ENA = 9;
const int IN1 = 8;
const int IN2 = 7;

void setup() {
  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  digitalWrite(IN1, HIGH); // Sens 1
  digitalWrite(IN2, LOW);
}

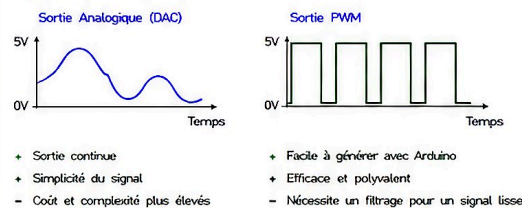
void loop() {
  for (int vitesse = 0; vitesse <= 255; vitesse += 5) {
    analogWrite(ENA, vitesse);
    delay(20);
  }
  for (int vitesse = 255; vitesse >= 0; vitesse -= 5) {
    analogWrite(ENA, vitesse);
    delay(20);
  }
}
```

Changez la direction en inversant IN1/IN2.

APPLICATIONS DU PWM

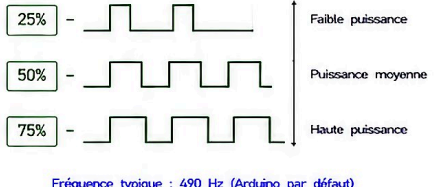
- Contrôle de vitesse des moteurs DC et pas à pas
- Variation de luminosité des LED
- Contrôle de servo moteurs
- Chauffage, ventilateurs, pompes, etc.

COMPARAISON : ANALOGIQUE VS PWM



- + Sortie continue
- + Simplicité du signal
- Coût et complexité plus élevés
- + Facile à générer avec Arduino
- + Efficace et polyvalent
- Nécessite un filtrage pour un signal lisse

SIGNAL PWM ET RAPPORT CYCLIQUE



BONNES PRATIQUES

- ✓ Utilisez le PWM pour un contrôle précis et fluide.
- ✓ Ne dépassez pas la tension et le courant max des moteurs et modules.
- ✓ Ajoutez des condensateurs de découplage (100 µF) sur l'alimentation pour réduire le bruit.
- ✓ Fixez bien le dissipateur thermique sur le L298N si vous utilisez de forts courants.
- ✓ Testez toujours progressivement avant d'atteindre la puissance maximale.

EXEMPLE DE PROJET : VOITURE TÉLÉCOMMANDÉE



- Contrôle à distance via Bluetooth
- Moteurs DC contrôlés par PWM
- Direction avant / arrière / stop
- Vitesse ajustable

Le PWM permet des mouvements fluides et un meilleur contrôle de votre robot.

À RETENIR

- ✓ Le PWM simule une tension analogique avec des impulsions numériques.
- ✓ analogWrite() permet de contrôler le rapport cyclique (0 à 255).
- ✓ Combinez PWM + pont en H (L298N) pour contrôler vitesse et direction des moteurs.
- ✓ Maîtriser le PWM améliore la précision et la performance de vos projets mécatroniques.

3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

3.26

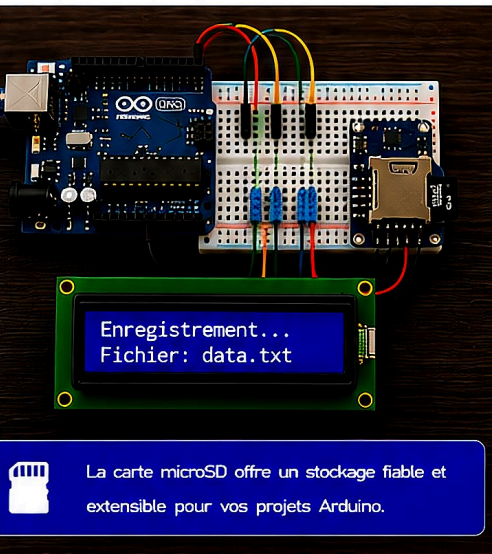
STOCKAGE ET CARTES MÉMOIRE AVEC ARDUINO

Les cartes mémoire permettent de stocker des données de façon permanente. Elles sont idéales pour l'enregistrement de logs, d'images ou de configurations.

Nous allons apprendre à utiliser une carte microSD avec Arduino pour lire et écrire des fichiers texte facilement.



Les données enregistrées restent disponibles même après la mise hors tension.



La carte microSD offre un stockage fiable et extensible pour vos projets Arduino.

TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interfaces	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.12	Stockage et cartes mémoire	71
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Projets avancés	76
3.19	Interruptions externes	77
3.20	Projets avancés avec interruptions	78
3.21	Actionneurs et sorties numériques	80
3.22	Capteurs et interfaces	81
3.23	Communication série et visualisation	82
3.24	PWM avancé et contrôle des moteurs	83
3.26	Stockage et cartes mémoire	84

1 MATÉRIEL NÉCESSAIRE

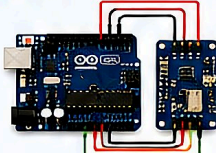
- Arduino Uno
- Module microSD (SPI)
- Carte microSD (formatée en FAT32)
- Fils de connexion

Connexion (SPI)

Module microSD

VCC → 5V
GND → GND
MOSI → D11
MISO → D12
SCK → D13
CS → D10

Arduino Uno



Always formatez la carte microSD en FAT32 pour assurer la compatibilité.

2 INCLURE LA BIBLIOTHÈQUE

Nous utilisons la bibliothèque SD pour interagir avec la carte microSD via le protocole SPI.

```
#include <SPI.h>
#include <SD.h>

File myFile;
const int CS_PIN = 10; // Chip Select
```

Étapes

1. Insérez la carte microSD.
2. Connectez les broches comme indiqué.
3. Incluez les bibliothèques SPI et SD.
4. Initialisez la carte avec SD.begin().



SPI est la méthode de communication la plus rapide pour les cartes mémoire.

3 INITIALISER LA CARTE SD

Vérifiez si la carte est détectée et prête à l'emploi.

```
void setup() {
  Serial.begin(9600);
  while (!Serial);
  Serial.println("Initialisation SD...");
  if (!SD.begin(CS_PIN)) {
    Serial.println("Échec de la carte SD!");
    while (1);
  }
  Serial.println("Carte SD prête.");
}
```



Si SD.begin() échoue, vérifiez le câblage et le formatage de la carte.

4 ÉCRIRE DANS UN FICHIER

Créons et écrivons des données dans un fichier texte.

```
void saveData() {
  myFile = SD.open("data.txt", FILE_WRITE);
  if (myFile) {
    myFile.println("DEX ATOMES");
    myFile.print("Température: ");
    myFile.print(25.6);
    myFile.println(" °C");
    myFile.close();
    Serial.println("Données enregistrées.");
  } else {
    Serial.println("Erreur d'ouverture!");
  }
}
```



FILE_WRITE ajoute les données à la fin du fichier sans supprimer le contenu existant.

5 LIRE UN FICHIER

Ouvrons le fichier et lisons son contenu ligne par ligne.

```
void readData() {
  myFile = SD.open("data.txt");
  if (myFile) {
    Serial.println("--- Contenu du fichier ---");
    while (myFile.available()) {
      String line = myFile.readStringUntil('\n');
      Serial.println(line);
    }
    myFile.close();
  } else {
    Serial.println("Fichier introuvable!");
  }
}
```



readStringUntil('\n') lit une ligne jusqu'au saut de ligne.

6 EXEMPLE COMPLET

Voici un exemple combinant écriture et lecture.

```
void loop() {
  static unsigned long last = 0;
  if (millis() - last > 5000) {
    saveData(); // Écrire
    delay(1000);
    readData(); // Lire
    Serial.println();
    last = millis();
  }
}
```



Ce programme enregistre des données toutes les 5 secondes et les affiche dans le moniteur série.



BONNES PRATIQUES

- ✓ Utilisez des noms de fichiers simples (8.3 compatible).
- ✓ Fermez toujours le fichier après utilisation (myFile.close()).
- ✓ Évitez les débranchements pendant l'écriture.
- ✓ Sauvegardez régulièrement les données importantes.
- ✓ Testez votre carte avec le moniteur série.

IDÉES DE PROJETS



Station météo
Enregistrez les mesures (température, humidité, pression).



Datalogger
Stockez les données des capteurs pour analyse ultérieure.



Caméra Arduino
Enregistrez des images ou des snapshots sur la carte SD.



À RETENIR

- ✓ Les cartes microSD permettent un stockage permanent et extensible.
- ✓ Utilisez la bibliothèque SD avec SPI.
- ✓ Les fichiers peuvent être créés, écrits, lus et supprimés.
- ✓ Toujours formater la carte en FAT32.
- ✓ Idéal pour les enregistreurs de données et les projets autonomes.

3

CHAPITRE 3

Arduino et ESP32
– Fondamentaux

3.27

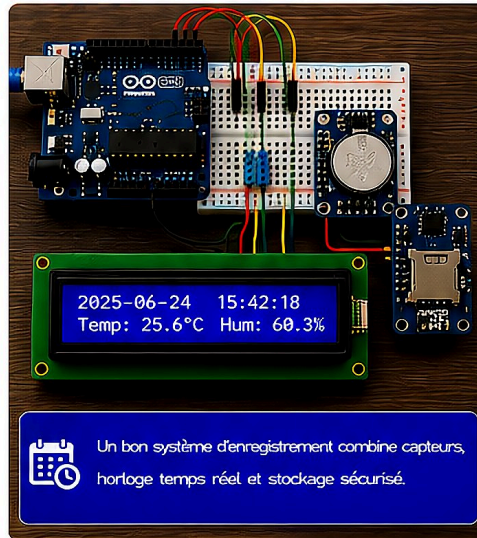
ENREGISTREMENT DE DONNÉES AVEC HORODATAGE

Ajouter une date et une heure à chaque mesure permet de suivre l'évolution des données dans le temps.

Nous utiliserons un module RTC (Real Time Clock) DS3231 avec une carte microSD pour créer un système de journalisation fiable et autonome.



L'horodatage précis est essentiel pour l'analyse des données et la prise de décision.



Un bon système d'enregistrement combine capteurs, horloge temps réel et stockage sécurisé.

TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Interruptions externes	76
3.19	Projets avancés avec interruptions	78
3.20	Matrices LED et animations	80
3.21	Actionneurs et sorties numériques	81
3.22	Capteurs et interfaces	82
3.23	Communication série et visualisation	83
3.24	Stockage et cartes mémoire	84
3.27	Enregistrement de données	85

1 MATÉRIEL NÉCESSAIRE

- Arduino Uno
- Module RTC DS3231 (I2C)
- Carte microSD (formatée FAT32)
- Capteur DHT22 (température/humidité)
- Afficheur LCD 16x2 (optionnel)
- Fils de connexion
- Breadboard



RTC DS3231



microSD Module



DHT22 Capteur



LCD 16x2



Le RTC garde l'heure même sans alimentation grâce à sa pile CR2032.

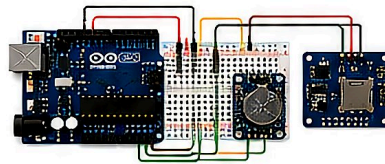
2 CONNEXIONS (SCHEMA I2C + SPI)

I2C (RTC DS3231)

SDA → A4
SCL → A5
VCC → 5V
GND → GND

SPI (microSD)

CS → D10
MOSI → D11
MISO → D12
SCK → D13
VCC → 5V
GND → GND



Assurez-vous que le bus SPI utilise les broches par défaut pour la compatibilité.

3 INCLURE LES BIBLIOTHÈQUES

```
#include <Wire.h> // I2C
#include <RTCLib.h> // RTC DS3231
#include <SD.h> // Carte SD
#include <SPI.h> // Communication SPI
#include <DHT.h> // Capteur DHT22
```

Définir les broches :

```
#define DHTPIN 2
#define DHTTYPE DHT22
#define SD_CS 10
```



La bibliothèque RTCLib simplifie la gestion de la date et de l'heure.

4 INITIALISER LES MODULES

```
RTC_DS3231 rtc;
DHT dht(DHTPIN, DHTTYPE);
File myFile;

void setup() {
  Serial.begin(9600);
  Wire.begin(); // I2C
  dht.begin(); // DHT22
  rtc.begin();

  if (!rtc.isrunning()) {
    Serial.println("RTC non démarrée !");
  }

  if (!SD.begin(SD_CS)) {
    Serial.println("Erreur carte SD !");
    while (1);
  }
}
```



Vérifiez que le RTC est à l'heure lors du premier démarrage.

5 ENREGISTRER LES DONNÉES

```
void loop() {
  DateTime now = rtc.now();
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  if (isnan(t) || isnan(h)) {
    Serial.println("Erreur lecture DHT22");
    delay(2000);
    return;
  }

  // Ouvre le fichier en mode écriture (ajout)
  myFile = SD.open("data.txt", FILE_WRITE);
  if (myFile) {
    myFile.print(now.timestamp(DateTime::TIMESTAMP_FULL));
    myFile.print(",");
    myFile.print(t, 1);
    myFile.print(",");
    myFile.print(h, 1);
    myFile.println(); // Ferme le fichier
  }
  delay(60000); // Attendre 1 minute
}
```



Chaque ligne contient : date, heure, température et humidité.

6 EXEMPLE DE FICHIER (data.txt)

Date et heure	Temp (°C)	Humidité (%)
2025-06-24 15:40:00	25.4	59.8
2025-06-24 15:41:00	25.5	60.1
2025-06-24 15:42:00	25.6	60.3
2025-06-24 15:43:00	25.7	60.4
2025-06-24 15:44:00	25.6	60.2

Ce fichier peut être ouvert avec Excel, Google Sheets ou tout éditeur de texte.



Importez le fichier dans Excel pour créer des graphiques et analyser les tendances.



BONNES PRATIQUES

- ✓ Utilisez une carte SD de qualité (classe 10).
- ✓ Formatez toujours la carte en FAT32.
- ✓ Vérifiez régulièrement l'heure du RTC.
- ✓ Ajoutez un en-tête dans le fichier pour faciliter l'analyse.
- ✓ Évitez d'ouvrir le fichier trop fréquemment si vous enregistrez souvent.
- ✓ Sauvegardez et transférez les données avant d'éteindre le système.



IDÉES DE PROJETS



Station météo
Enregistrez température, humidité, pression et luminosité.



Serre intelligente
Surveillez l'environnement pour optimiser la croissance des plantes.



Niveau d'eau
Mesurez et enregistrez le niveau d'eau dans un réservoir.



À RETENIR

- ✓ Le RTC DS3231 fournit un horodatage précis et fiable.
- ✓ La carte microSD permet un stockage autonome et extensible.
- ✓ Le formatage FAT32 assure la compatibilité.
- ✓ L'enregistrement régulier donne une vision claire des variations.
- ✓ Combinez capteurs + horodatage + stockage pour des projets professionnels.

3

CHAPITRE 3

Arduino et ESP32

- Fondamentaux

3.28

AFFICHAGE LCD 16x2 AVEC ARDUINO

Les écrans LCD 16x2 permettent d'afficher du texte sur deux lignes de 16 caractères. Ils sont très utilisés pour les interfaces homme-machine simples.

Dans cette section, nous apprenons à connecter un écran LCD 16x2 avec un module I2C pour réduire le nombre de broches utilisées et afficher des informations de manière claire.



L'utilisation d'un module I2C simplifie le câblage : seulement 4 fils nécessaires !

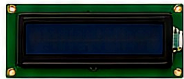


TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Affichages LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Interruptions externes	76
3.19	Projets avancés avec interruptions	78
3.20	Matrices LED et animations	80
3.21	Actionneurs et sorties numériques	81
3.22	Capteurs et interfaces	82
3.23	Communication série et visualisation	83
3.24	Stockage et cartes mémoire	84
3.28	Affichage LCD 16x2 avec Arduino	86

1 MATÉRIEL NÉCESSAIRE

- Arduino Uno
- Écran LCD 16x2
- Module I2C (PCF8574)
- Fils de connexion
- Breadboard (optionnel)



LCD 16x2



Module I2C (PCF8574)



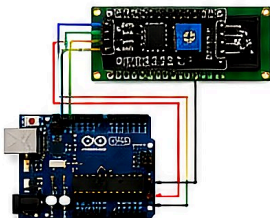
Arduino Uno



Le module I2C se connecte à l'arrière de l'écran LCD.

2 SCHÉMA DE CONNEXION (I2C)

LCD I2C (PCF8574)	Arduino Uno
VCC	5V
GND	GND
SDA	A4 (SDA)
SCL	A5 (SCL)



Les broches SDA et SCL utilisent le bus I2C de l'Arduino.

3 INSTALLER LA BIBLIOTHÈQUE

Nous utilisons la bibliothèque LiquidCrystal_I2C.

Étapes :

1. Ouvrez l'IDE Arduino.
2. Allez dans Outils > Gérer les bibliothèques.
3. Recherchez LiquidCrystal I2C.
4. Installez la bibliothèque de Frank de Brabander.
5. Incluez dans votre code :

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



Redémarrez l'IDE après l'installation si nécessaire.

4 EXEMPLE DE CODE

```
#include <LiquidCrystal_I2C.h>

// Adresse I2C (0x27 ou 0x3F)
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  lcd.init();           // Initialise l'écran
  lcd.backlight();      // Allume le rétroéclairage
  lcd.setCursor(0, 0);
  lcd.print("Dex Atomes");
  lcd.setCursor(0, 1);
  lcd.print("LCD 16x2 I2C");
}

void loop() {
  // Rien à faire ici
}
```

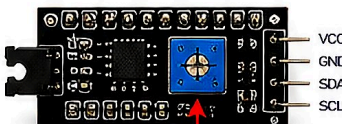


Si rien ne s'affiche, essayez l'adresse 0x3F au lieu de 0x27.

OPTIONS UTILES

1 CHANGER LE CONTRASTE

Le module I2C possède un potentiomètre (petite vis bleue) pour régler le contraste de l'écran.



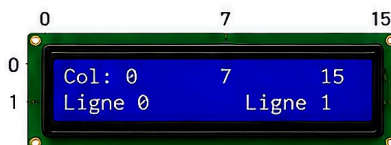
Régler le contraste

2 POSITION DU CURSEUR

Les lignes et colonnes commencent à 0.

Exemples :

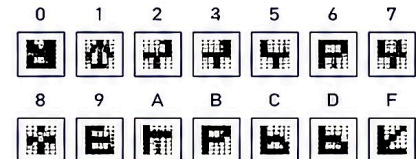
- `setCursor(0,0)` → première ligne, première colonne
- `setCursor(15,0)` → première ligne, dernière colonne
- `setCursor(0,1)` → deuxième ligne, première colonne



3 CARACTÈRES SPÉCIAUX

Vous pouvez afficher des caractères personnalisés.

Exemples intégrés :



Voir la documentation LiquidCrystal_I2C pour plus de détails.

BONNES PRATIQUES

- ✓ Réglez bien le contraste pour une lecture optimale.
- ✓ Utilisez le module I2C pour économiser les broches.
- ✓ Évitez d'écrire trop fréquemment sur l'écran (cela cause un clignotement).
- ✓ Effacez uniquement quand nécessaire avec `lcd.clear()`.
- ✓ Testez votre affichage étape par étape.

IDÉES DE PROJETS



Thermomètre numérique
Affichez la température et l'humidité en temps réel.



Horloge numérique
Affichez l'heure et la date de votre module RTC.



Menu interactif
Créez des menus simples à l'aide de boutons.

À RETENIR

- ✓ L'écran LCD 16x2 est idéal pour afficher des textes courts.
- ✓ Le module I2C (PCF8574) simplifie grandement le câblage.
- ✓ Les fonctions de base : `init()`, `backlight()`, `setCursor()`, `print()`, `clear()`.
- ✓ Toujours vérifier l'adresse I2C (0x27 ou 0x3F).
- ✓ L'affichage améliore l'interface utilisateur de vos projets.

3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

3.29

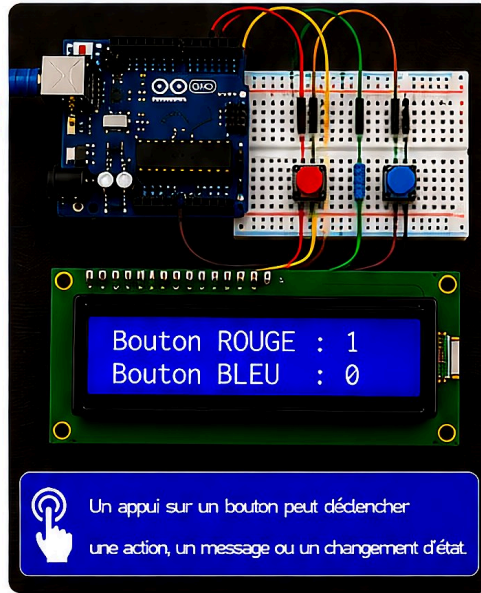
BOUTONS POUSSOIRS ET INTERFACES UTILISATEUR AVEC ARDUINO

Les boutons poussoirs permettent d'interagir avec un système électronique en envoyant des commandes simples à l'Arduino.

Nous apprendrons à lire l'état d'un bouton, à gérer un appui et un relâchement, et à afficher le résultat sur un écran LCD 16x2.



Les interfaces utilisateur simples améliorent l'expérience et le contrôle de vos projets !



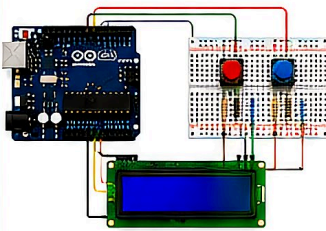
Un appui sur un bouton peut déclencher une action, un message ou un changement d'état.

TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	64
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Interruptions externes	76
3.19	Projets avancés avec interruptions	78
3.20	Matrices LED et animations	80
3.21	Actionneurs et sorties numériques	81
3.22	Capteurs et interfaces	82
3.23	Communication série et visualisation	83
3.24	Stockage et cartes mémoire	84
3.25	Enregistrement de données	85
3.26	Affichage LCD 16x2 avec Arduino	86
3.29	Boutons poussoirs et interfaces	87

1 MATÉRIEL NÉCESSAIRE

- Arduino Uno
- 2 boutons poussoirs (ROUGE et BLEU)
- 2 résistances 10 kΩ (pull-down)
- Écran LCD 16x2 avec I2C
- Fils de connexion
- Breadboard



D2 → Bouton ROUGE
D3 → Bouton BLEU

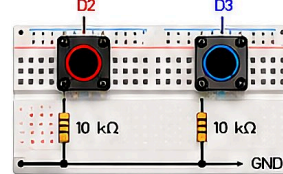


Les résistances pull-down évitent les lectures erronées (bruit).

2 SCHÉMA DE CONNEXION

Élément	Connexion Arduino
Bouton ROUGE	→ D2
Bouton BLEU	→ D3
Résistance 10 kΩ (ROUGE)	→ GND
Résistance 10 kΩ (BLEU)	→ GND
Écran LCD 16x2 (I2C)	A4 (SDA) et A5 (SCL)
VCC (LCD)	→ 5V
GND (LCD)	→ GND

Organisation des boutons :



Le bouton est connecté entre 5V et la broche, la résistance tire la broche à GND par défaut.

3 CODE : LIRE LES BOUTONS

```
const int btnRouge = 2; // Bouton ROUGE
const int btnBleu = 3; // Bouton BLEU

void setup() {
  pinMode(btnRouge, INPUT_PULLUP); // ou INPUT
  pinMode(btnBleu, INPUT_PULLUP);
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0,0);
  lcd.print("Dex Atomes");
}

void loop() {
  int etatRouge = digitalRead(btnRouge);
  int etatBleu = digitalRead(btnBleu);

  lcd.setCursor(0,1);
  lcd.print("R: "); lcd.print(etatRouge);
  lcd.print(" B: "); lcd.print(etatBleu);
  delay(100); // petite stabilité
}
```



INPUT_PULLUP active la résistance interne au 5V. Appui = 0 (LOW), Relâchement = 1 (HIGH).

4 AFFICHAGE SUR L'ÉCRAN

Aucun bouton appuyé :



Bouton ROUGE appuyé :



Bouton BLEU appuyé :



L'état 0 indique un appui (bouton connecté à 5V via le bouton).

5 AMÉLIORATION : COMPTER LES APPUIS

```
int compteurRouge = 0;
int compteurBleu = 0;

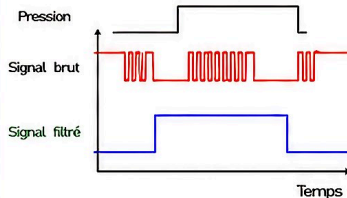
void loop() {
  if (digitalRead(btnRouge) == LOW) {
    compteurRouge++;
    delay(200); // anti-rebond simple
  }
  if (digitalRead(btnBleu) == LOW) {
    compteurBleu++;
    delay(200);
  }
  lcd.setCursor(0,0);
  lcd.print("R: "); lcd.print(compteurRouge);
  lcd.print(" B: "); lcd.print(compteurBleu);
}
```



Le délai évite de compter plusieurs fois le même appui (rebond mécanique).

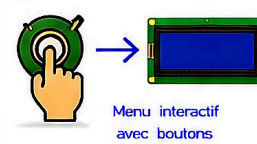
6 ANTI-REBOND AVEC BOUTON

Pour une meilleure fiabilité, utilisez une bibliothèque ou un délai de stabilisation.



Le rebond peut générer plusieurs lectures rapides indésirables.

7 IDÉE DE PROJET



- Naviguer dans un menu
- Régler des paramètres
- Afficher des informations



Combinez boutons + écran pour un contrôle simple et efficace.

8 VARIANTES

- ✓ Boutons normalement ouverts (NO) ou fermés (NC)
- ✓ Utiliser des résistances pull-up externes (10 kΩ)
- ✓ Détecter appui long ou double appui
- ✓ Ajouter un buzzer pour confirmer les actions

BONNES PRATIQUES

- ✓ Utilisez INPUT_PULLUP pour simplifier le câblage.
- ✓ Ajoutez un délai contre le rebond des boutons.
- ✓ Testez chaque bouton individuellement.
- ✓ Commentez votre code pour plus de clarté.
- ✓ Organisez votre code pour une évolutivité facile.

IDÉES D'APPLICATIONS



Interfaces homme-machine (IHM)



Contrôle de menu sur écran LCD



Systèmes d'alarme ou de sécurité



Jeux ou quiz interactifs

À RETENIR

- ✓ Les boutons poussoirs envoient un signal logique à l'Arduino.
- ✓ Utilisez les résistances pull-down ou pull-up.
- ✓ L'écran LCD permet d'afficher les états clairement.
- ✓ Une bonne interface utilisateur rend vos projets plus conviviaux !

3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

3.30

BUZZER ET SONNERIES AVEC ARDUINO

Le buzzer permet de générer des sons ou des sonneries pour alerter, notifier ou créer des effets sonores.

Nous utiliserons un buzzer passif pour produire différentes fréquences et créer des mélodies simples à l'aide de la fonction `tone()` d'Arduino.



Le son est produit en envoyant un signal carré (PWM) à une fréquence spécifique au buzzer.



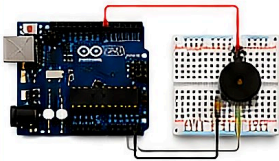
Le buzzer transforme les signaux électriques en sons audibles pour vos projets !

TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Interruptions externes	76
3.19	Projets avancés avec interruptions	78
3.20	Matrices LED et animations	80
3.21	Actionneurs et sorties numériques	81
3.22	Capteurs et interfaces	82
3.23	Communication série et visualisation	83
3.24	Stockage et cartes mémoire	84
3.25	Enregistrement de données	85
3.26	Affichage LCD 16x2 avec Arduino	86
3.27	Boutons poussoirs et interfaces	87
3.30	Buzzer et sonneries avec Arduino	88

1 MATÉRIEL NÉCESSAIRE

- Arduino Uno
- Buzzer passif
- Résistance 220 Ω (optionnelle)
- Fils de connexion
- Breadboard



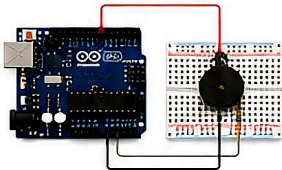
D8 → Signal (PWM)
GND → Masse



La résistance protège le buzzer en limitant le courant.

2 SCHÉMA DE CONNEXION

Buzzer	Connexion Arduino
+ (positif)	D8 (PWM)
- (négatif)	GND
(Résistance)	En série avec le +



Le buzzer passif n'a pas de polarité, mais respectez le schéma pour éviter les erreurs.

3 CODE : TON ET DURÉE

```
// --- Buzzer ---
const int buzzer = 8;

void setup() {

}

void loop() {
  tone(buzzer, 440, 500); // 440 Hz
  delay(600);             // pause
  tone(buzzer, 660, 500); // 660 Hz
  delay(600);
  tone(buzzer, 784, 500); // 784 Hz
  delay(600);
  noTone(buzzer);         // stop son
  delay(1000);            // attente
}
```



`tone(pin, fréquence, durée_ms)` génère un son à une fréquence donnée pendant une durée.

4 FRÉQUENCES MUSICALES (EXEMPLES)

Note	Fréquence (Hz)	Durée (ms)
Do (C)	261	500
Ré (D)	294	500
Mi (E)	329	500
Fa (F)	349	500
Sol (G)	392	500
La (A)	440	500
Si (B)	494	500
Do (C5)	523	500



Ces fréquences sont des références pour la gamme tempérée (A4 = 440 Hz).

5 CRÉER UNE MÉLODIE SIMPLE

Voici un exemple de mélodie :

```
int notes[] = { 262, 262, 294, 294, 330, 330, 294, 0,
               262, 262, 294, 294, 330, 330, 294, 0 };
int durees[] = { 4, 4, 4, 4, 4, 4, 2, 4,
                 4, 4, 4, 4, 4, 4, 2, 4 };

void loop() {
  for (int i = 0; i < 16; i++) {
    int duree = 1000 / durees[i];
    tone(buzzer, notes[i], duree);
    delay(duree * 1.30); // petite pause entre les notes
    noTone(buzzer);
  }
  while (1); // répète la mélodie
}
```



0 dans le tableau signifie une pause (silence).

6 DIFFÉRENCE : PASSIF VS ACTIF

Buzzer passif

- Pas de circuit interne
- Nécessite un signal alternatif (`tone()`)
- Peut jouer des notes et mélodies



Buzzer actif

- Circuit interne (oscillateur)
- Émet un son avec un niveau logique HIGH
- Ne peut pas jouer de mélodies



Pour les mélodies, utilisez toujours un buzzer passif !

7 EXEMPLE : ALARME SIMPLE

Le buzzer émet un bip rapide en cas d'alarme.

```
void loop() {
  // Alarme : 3 bips rapides
  for (int i = 0; i < 3; i++) {
    tone(buzzer, 1000, 150);
    delay(200);
  }
  delay(2000); // attente avant de recommencer
}
```



Utilisez des bips pour des alertes, des minuteurs ou des notifications.



BONNES PRATIQUES

- ✓ Utilisez `tone()` et `noTone()` pour contrôler le son.
- ✓ Évitez des fréquences trop basses (< 100 Hz) ou trop hautes (> 5000 Hz).
- ✓ Ajoutez des pauses entre les notes pour de meilleures mélodies.
- ✓ Si le son est faible, vérifiez la connexion et l'alimentation.



IDÉES DE PROJETS



Sonnerie intelligente

Joue un son quand quelqu'un appuie sur un bouton.



Alerte de distance

Bip plus rapide quand un objet est trop proche.



Minuteur de cuisine

Bip à la fin du temps réglé.



À RETENIR

- ✓ Le buzzer transforme les signaux électriques en sons audibles.
- ✓ `tone()` permet de générer des notes musicales.
- ✓ Un buzzer passif est idéal pour les mélodies.
- ✓ Utilisez le son pour alerter, notifier ou améliorer l'expérience utilisateur.

3

CHAPITRE 3 Arduino et ESP32 - Fondamentaux

3.31

LED ET 7 SEGMENTS AVEC ARDUINO

Les LED et les afficheurs 7 segments sont très utiles pour afficher des états, des compteurs ou des informations numériques.

Dans cette section, nous contrôlons une LED et un afficheur 7 segments pour afficher les chiffres de 0 à 9.



Les LED consomment peu d'énergie et les afficheurs 7 segments permettent d'afficher des nombres de manière claire et efficace.



La LED indique un état et l'afficheur 7 segments affiche des chiffres lisibles par tous.

TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Interruptions externes	76
3.19	Projets avancés avec interruptions	78
3.20	Matrices LED et animations	80
3.21	Actionneurs et sorties numériques	81
3.22	Capteurs et interfaces	82
3.23	Communication série et visualisation	83
3.24	Stockage et cartes mémoire	84
3.25	Enregistrement de données	85
3.26	Affichage LCD 16x2 avec Arduino	86
3.27	Boutons poussoirs et interfaces	87
3.30	Buzzer et sonneries avec Arduino	88
3.31	LED et 7 segments avec Arduino	89

1 MATÉRIEL NÉCESSAIRE

- Arduino Uno
- 1 LED rouge
- Résistance 220 Ω
- Afficheur 7 segments (cathode commune)
- Résistances 220 Ω (7 pièces)
- Fils de connexion
- Breadboard



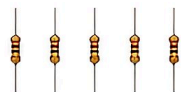
Arduino Uno



LED + 220 Ω



7 segments (CC)



220 Ω (x7)

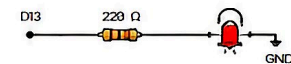


La cathode commune (CC) se connecte toujours à la masse (GND).

2 SCHÉMA DE CONNEXION

LED

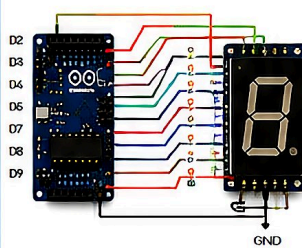
D13 → Résistance 220 Ω → Anode LED



Afficheur 7 segments (cathode commune)

Segment	Broche Arduino	Segment	Broche Arduino
a	D2	e	D6
b	D3	f	D7
c	D4	g	D8
d	D5	dp	D9

Cathode commune → GND



Chaque segment doit avoir sa propre résistance pour limiter le courant.

3 CODE : LED ET COMPTEUR 0 À 9

```
// LED sur D13
const int led = 13;

// Segments 7 segments (a,b,c,d,e,f,g,dp)
const int seg[8] = {2, 3, 4, 5, 6, 7, 8, 9};

// Table des chiffres 0 à 9 (CC)
// a b c d e f g dp
byte chiffres[10][8] = {
  {1,1,1,1,1,1,0,0}, // 0
  {0,1,1,0,0,0,0,0}, // 1
  {1,1,0,1,1,0,1,0}, // 2
  {1,1,1,1,0,0,1,0}, // 3
  {0,1,1,0,0,1,1,0}, // 4
  {1,0,1,1,0,1,1,0}, // 5
  {1,0,1,1,1,1,1,0}, // 6
  {1,1,1,0,0,0,0,0}, // 7
  {1,1,1,1,1,1,1,0}, // 8
  {1,1,1,1,0,1,1,0}, // 9
};

void setup() {
  pinMode(led, OUTPUT);
  for (int i=0; i<8; i++) pinMode(seg[i], OUTPUT);
}

void loop() {
  for (int n=0; n<10; n++) {
    digitalWrite(led, HIGH); // LED ON
    afficherChiffre(n);
    delay(1000);
    digitalWrite(led, LOW); // LED OFF
    delay(200);
  }
}

void afficherChiffre(int n) {
  for (int i=0; i<8; i++)
    digitalWrite(seg[i], chiffres[n][i]);
}
```



La LED clignote et l'afficheur compte de 0 à 9 en boucle.

4 AFFICHAGE DES CHIFFRES

Chiffre	Segments ALLUMÉS (1)						
	a	b	c	d	e	f	g
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1

Exemple visuel :



La luminosité peut être ajustée en utilisant la PWM au lieu de HIGH/LOW.



BONNES PRATIQUES

- ✓ Utilisez toujours une résistance pour la LED.
- ✓ Limitez le courant des segments avec 220 Ω.
- ✓ Vérifiez le type : cathode commune (CC).
- ✓ Organisez votre code avec des fonctions.
- ✓ Testez chaque composant individuellement.
- ✓ Commentez votre code pour plus de clarté.
- ✓ Évitez d'utiliser trop de broches si possible.

IDÉES DE PROJETS



Compteur d'événements

Affiche le nombre de clics ou d'événements.



Minuteur numérique

Compte à rebours avec alarme sonore.



Dé électronique

Génère un nombre aléatoire (1 à 6).



À RETENIR

- ✓ La LED permet d'indiquer un état simple.
- ✓ L'afficheur 7 segments affiche des chiffres clairs et lisibles.
- ✓ Chaque segment nécessite une résistance.
- ✓ Utilisez des tableaux pour simplifier le code.
- ✓ Ces bases sont essentielles pour vos futurs projets électroniques.

3

CHAPITRE 3 Arduino et ESP32 – Fondamentaux

3.32

POTENTIOMÈTRE ET LECTURE ANALOGIQUE AVEC ARDUINO

Le potentiomètre est un composant variable qui permet de régler une tension entre 0V et 5V.
Nous allons lire sa valeur avec Arduino et l'afficher sur l'écran LCD 16x2.



Les entrées analogiques permettent à Arduino de mesurer des tensions variables et de réagir en conséquence.



Le potentiomètre transforme un mouvement rotatif en une tension variable que l'Arduino peut mesurer.

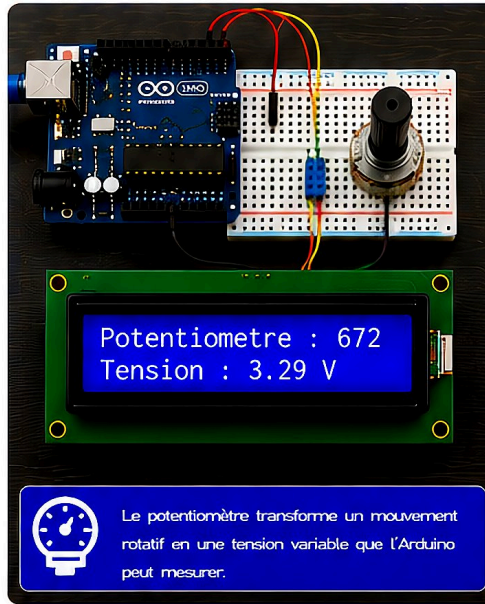


TABLE DES MATIÈRES

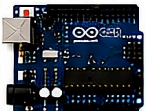
3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Interruptions externes	76
3.19	Projets avancés avec interruptions	78
3.20	Matrices LED et animations	80
3.21	Actionneurs et sorties numériques	81
3.22	Capteurs et interfaces	82
3.23	Communication série et visualisation	83
3.24	Stockage et cartes mémoire	84
3.25	Enregistrement de données	85
3.26	Affichage LCD 16x2 avec Arduino	86
3.27	Boutons poussoirs et interfaces	87
3.29	LED et 7 segments avec Arduino	89
3.32	Potentiomètre et lecture analogique	90

1 MATÉRIEL NÉCESSAIRE

- Arduino Uno
- Potentiomètre 10 kΩ
- Résistance 220 Ω (optionnelle)
- Afficheur LCD 16x2 avec I2C
- Fils de connexion
- Breadboard



Potentiomètre 10 kΩ



Arduino Uno



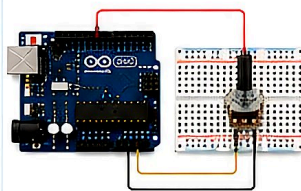
LCD 16x2 I2C



Le potentiomètre possède 3 broches : GND, VCC et SIGNAL (curseur).

2 SCHÉMA DE CONNEXION

Potentiomètre	Connexion Arduino
VCC (extrémité 1)	→ 5V
SIGNAL (curseur)	→ A0
GND (extrémité 2)	→ GND



Le curseur (broche centrale) envoie une tension variable entre 0V et 5V vers A0.

3 CODE : LECTURE ANALOGIQUE

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

LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  lcd.init();
  lcd.backlight();
  lcd.clear();
}

void loop() {
  int valeur = analogRead(A0);
  float tension = valeur * (5.0 / 1023.0);

  lcd.setCursor(0, 0);
  lcd.print("Potentiometre : ");
  lcd.print(valeur);
  lcd.setCursor(0, 1);
  lcd.print("Tension : ");
  lcd.print(tension, 2);
  lcd.print(" V");
  delay(200);
}
```



analogRead(A0) retourne une valeur entre 0 et 1023 (10 bits).

4 EXEMPLE DE SORTIE

Curseur à gauche (minimum)



Position intermédiaire

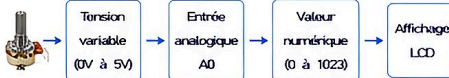


Curseur à droite (maximum)



La valeur analogique change en temps réel avec la position du curseur.

5 COMMENT ÇA FONCTIONNE ?



L'Arduino convertit la tension analogique en une valeur numérique grâce à son convertisseur analogique-numérique (ADC).



Plus la tension est élevée, plus la valeur numérique est grande.

6 UTILISATIONS COURANTES

- ✓ Régler la vitesse d'un moteur
- ✓ Ajuster la luminosité d'une LED
- ✓ Contrôler le volume d'un son
- ✓ Naviguer dans un menu
- ✓ Régler une consigne ou une valeur cible



Les entrées analogiques rendent vos projets beaucoup plus interactifs !

7 AMÉLIORER LA LECTURE (FILTRAGE)

Pour éviter les variations rapides (bruit), on peut faire la moyenne de plusieurs lectures.

```
int lectureMoyenne(int pin) {
  long somme = 0;
  for (int i = 0; i < 10; i++) {
    somme += analogRead(pin);
    delay(5);
  }
  return somme / 10;
}
```



Utiliser la moyenne permet d'obtenir une valeur plus stable et précise.



BONNES PRATIQUES

- ✓ Connectez correctement le potentiomètre.
- ✓ Vérifiez que la broche SIGNAL est reliée à A0.
- ✓ Évitez les fils trop longs qui captent du bruit.
- ✓ Utilisez le filtrage si la valeur fluctue trop.
- ✓ Testez toute la plage de 0 à 1023.



IDÉES DE PROJETS



Vitesse du robot

Régler la vitesse des moteurs avec le potentiomètre.



Variateur de lumière

Ajuster la luminosité d'une LED ou d'une bande LED.



Contrôle du volume

Modifier le volume d'un buzzer ou d'un module audio.



À RETENIR

- ✓ Le potentiomètre fournit une tension variable.
- ✓ Arduino lit cette tension via une entrée analogique.
- ✓ La valeur retournée est comprise entre 0 et 1023.
- ✓ Cette valeur peut être affichée, stockée ou utilisée pour contrôler d'autres composants.
- ✓ Les entrées analogiques sont essentielles pour des projets interactifs et dynamiques.

3

CHAPITRE 3

Arduino et ESP32

– Fondamentaux

3.33

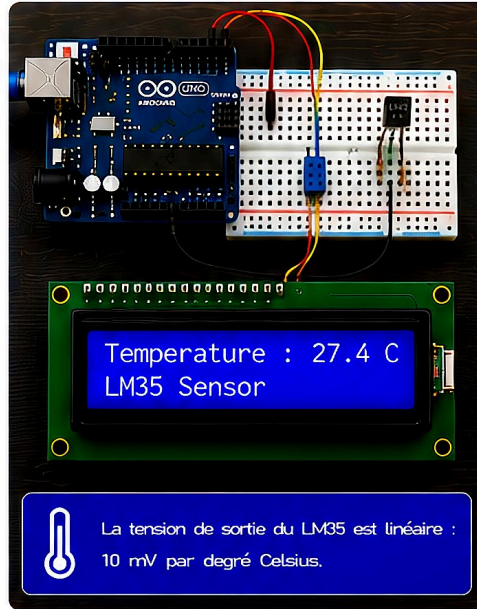
CAPTEUR DE TEMPÉRATURE LM35 AVEC ARDUINO

Le LM35 est un capteur de température analogique qui fournit une tension proportionnelle à la température en °C.

Nous lisons la température avec Arduino et l'affichons sur l'écran LCD 16x2.



Le LM35 est précis, facile à utiliser et ne nécessite pas de calibration externe.



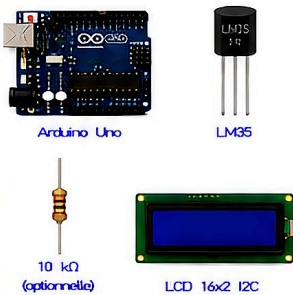
La tension de sortie du LM35 est linéaire : 10 mV par degré Celsius.

TABLE DES MATIÈRES

3.1	Introduction à Arduino	58
3.2	Carte Arduino Uno	59
3.3	ESP32 et microcontrôleurs	60
3.4	Entrées et sorties numériques	61
3.5	Entrées analogiques	62
3.6	Sorties PWM	63
3.7	Communication série UART	64
3.8	Communication I2C	65
3.9	Capteurs et interface	66
3.10	Actionneurs et sorties avancées	67
3.11	Afficheurs LCD	68
3.12	Matrices LED et 7 segments	69
3.13	Stockage et cartes mémoire	71
3.14	Horloge temps réel (RTC)	72
3.15	Projets pratiques	73
3.16	Communication série UART	74
3.17	Capteurs et interfaces	75
3.18	Interruptions externes	76
3.19	Projets avancés avec interruptions	78
3.20	Matrices LED et animations	80
3.21	Actionneurs et sorties numériques	81
3.22	Capteurs et interfaces	82
3.23	Communication série et visualisation	83
3.24	Stockage et cartes mémoire	84
3.25	Enregistrement de données	85
3.26	Affichage LCD 16x2 avec Arduino	86
3.27	Boutons poussoirs et interface	87
3.29	LED et 7 segments avec Arduino	89
3.30	Potentiomètre et lecture analogique	90
3.33	Capteur de température LM35	91

1 MATÉRIEL NÉCESSAIRE

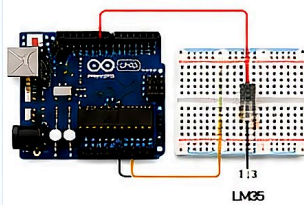
- Arduino Uno
- Capteur de température LM35
- Résistance 10 kΩ (optionnelle)
- Écran LCD 16x2 avec I2C
- Fils de connexion
- Breadboard



Le LM35 possède 3 broches : Vs (5V), Vout (signal), GND.

2 SCHÉMA DE CONNEXION

LM35	Connexion Arduino
Vs (broche 1)	→ 5V
Vout (broche 2)	→ A0
GND (broche 3)	→ GND



Ajoutez une résistance 10 kΩ entre Vout et GND pour stabiliser la lecture (optionnelle).

3 CODE : LECTURE DU LM35

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

LiquidCrystal_I2C lcd(0x27, 16, 2);

const int lm35Pin = A0;

void setup() {
  lcd.init();
  lcd.backlight();
  lcd.clear();
}

void loop() {
  int lecture = analogRead(lm35Pin);
  float tension = lecture * (5.0 / 1023.0);
  float temperature = tension * 100.0; // 10 mV/°C

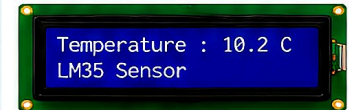
  lcd.setCursor(0, 0);
  lcd.print("Temperature : ");
  lcd.print(temperature, 1);
  lcd.print(" C ");

  lcd.setCursor(0, 1);
  lcd.print("LM35 Sensor");
  delay(1000);
}
```

La formule : Température (°C) = Tension (V) × 100.

4 RÉSULTAT SUR L'ÉCRAN

Exemple d'affichage à différentes températures :



Température basse



Température ambiante



Température élevée

La valeur affichée se met à jour toutes les 1 seconde.

5 COMMENT ÇA FONCTIONNE ?

Le LM35 génère une tension de 10 mV par degré Celsius.



Exemple :

- 0 °C → 0 mV
- 25 °C → 250 mV (0.25 V)
- 100 °C → 1000 mV (1.00 V)

L'Arduino lit cette tension via l'entrée analogique A0, la convertit en température et l'affiche.

6 PLAGE ET PRÉCISION

- Plage de mesure : -55 °C à +150 °C
- Précision typique : ±0.5 °C à 25 °C
- Linéarité : excellente (pas besoin de calibration)



Idéal pour des mesures simples et fiables.

7 ASTUCES PRATIQUES

- ✓ Placez le capteur loin des sources de chaleur (USB, régulateur).
- ✓ Utilisez des fils courts pour réduire le bruit.
- ✓ Ajoutez un condensateur 0.1 µF entre Vout et GND si la lecture fluctue.
- ✓ Vous pouvez convertir la température en Fahrenheit : °F = (°C × 9/5) + 32

8 IDÉES D'UTILISATION

- ☁ Station météo
- 🌡 Contrôle de température ambiante
- 🌀 Systèmes de ventilation
- 📢 Alarmes de surchauffe
- 🔋 Surveillance de batterie / moteur

BONNES PRATIQUES

- ✓ Vérifiez toujours les connexions du LM35 (1-2-3).
- ✓ Lisez plusieurs fois et prenez la moyenne pour plus de stabilité.
- ✓ Testez la plage complète du capteur.
- ✓ Commentez votre code pour plus de clarté.
- ✓ Utilisez des noms de variables explicites.

IDÉES DE PROJETS



Ventilateur automatique
Allume le ventilateur si la température dépasse un seuil.



Thermostat DIY
Contrôle un relais pour chauffer ou refroidir une pièce.



Enregistreur de données
Enregistrez les températures sur carte SD pour analyse.

À RETENIR

- ✓ Le LM35 est un capteur analogique simple et fiable.
- ✓ La sortie est linéaire : 10 mV par °C.
- ✓ Facile à interfacer avec Arduino.
- ✓ Idéal pour de nombreux projets de mesure et de contrôle.

3

CHAPITRE 3

Arduino
et ESP32



La lumière peut être mesurée et utilisée pour automatiser des systèmes intelligents.

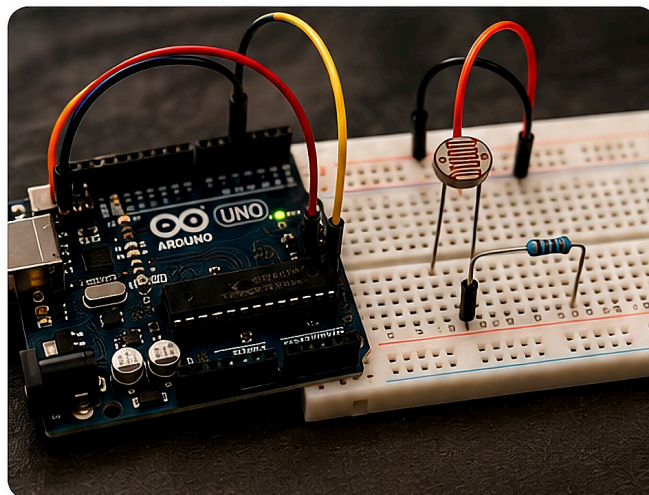
3.11

CAPTEUR DE LUMIÈRE (LDR) AVEC ARDUINO

Mesurer l'intensité lumineuse

Une LDR (Light Dependent Resistor) est une résistance dont la valeur varie selon la quantité de lumière reçue. Plus la lumière est forte, plus sa résistance diminue.

Les LDR sont utilisées dans l'éclairage automatique, les lampes intelligentes, les systèmes de sécurité et les robots autonomes.



MATÉRIEL NÉCESSAIRE

- Arduino Uno 1
- Capteur LDR 1
- Résistance 10 kΩ 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1

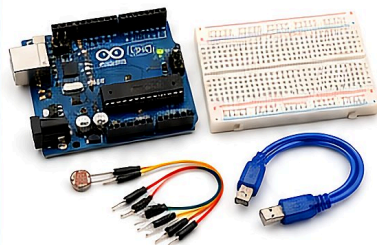
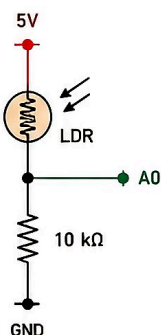


SCHÉMA DE CÂBLAGE

La LDR et la résistance forment un pont diviseur de tension. Arduino lit la tension au point A0.



- LDR → 5V
- LDR → A0
- Résistance 10 kΩ entre A0 et GND

CODE ARDUINO

```
int capteurLumiere = A0;
int valeurLumiere = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  valeurLumiere = analogRead(capteurLumiere);

  Serial.print("Luminosité : ");
  Serial.println(valeurLumiere);

  delay(500);
}
```



Ouvrez le Moniteur Série à 9600 bauds pour voir les valeurs en temps réel.

EXPLICATION DU PROGRAMME

La fonction `analogRead(A0)` mesure la tension issue du pont diviseur formé par la LDR et la résistance.

L'Arduino retourne une valeur numérique :

- 0 = très sombre
- 1023 = très lumineux

COMMENT FONCTIONNE UNE LDR

La résistance d'une LDR varie selon la lumière.



Forte lumière

Lumière moyenne

Obscurité

Faible résistance

Résistance moyenne

Résistance élevée

APPLICATION RÉELLE

Un lampadaire intelligent :



Le jour → lumière suffisante
→ lampe éteinte.



La nuit → faible luminosité
→ lampe allumée automatiquement.

Ce principe est utilisé dans des milliers de systèmes d'éclairage urbains modernes.



RÉSULTAT ATTENDU

Éclairez ou couvrez la LDR et observez les valeurs affichées dans le Moniteur Série.



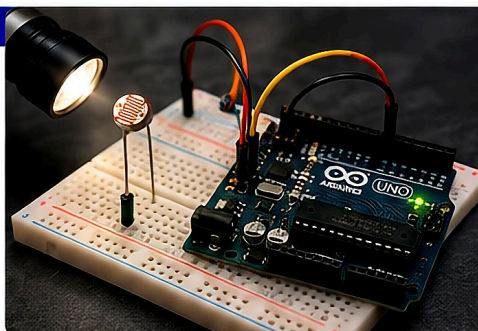
Exposé à la lumière
La valeur augmente
(ex. 800 - 1023)



Couvert / Ombre
La valeur diminue
(ex. 0 - 200)

COM3
Luminosité : 923
Luminosité : 918
Luminosité : 905

COM3
Luminosité : 123
Luminosité : 98
Luminosité : 87



À RETENIR

- ✓ Une LDR mesure l'intensité lumineuse.
- ✓ Sa résistance varie selon la lumière reçue.
- ✓ Arduino lit cette variation grâce à une entrée analogique.
- ✓ Les capteurs de lumière sont essentiels dans les systèmes automatisés et les robots intelligents.



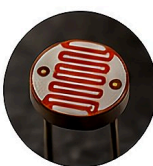
IDÉES D'AMÉLIORATION

- Allumer une LED automatiquement quand la lumière est faible.
- Afficher la luminosité sur un écran LCD.
- Enregistrer les données dans une carte SD.
- Combiner avec d'autres capteurs (PIR, température).



CONSEIL PRATIQUE

La valeur de la LDR peut varier selon l'environnement. Calibrez votre système en testant dans différentes conditions de lumière pour obtenir des résultats précis.



BON À SAVOIR

Les LDR ne détectent pas la couleur de la lumière. Elles mesurent uniquement l'intensité lumineuse globale (visible, infrarouge, ultraviolet).



3

CHAPITRE 3

Arduino
et ESP32



Le potentiomètre est un composant variable qui agit comme un diviseur de tension réglable.

3.12

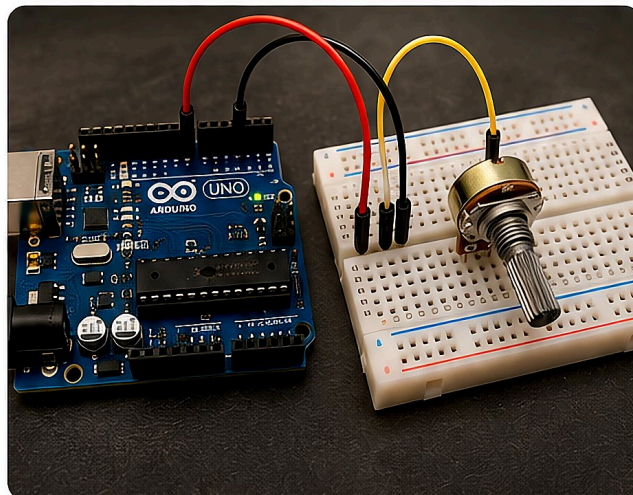
POTENTIOMÈTRE AVEC ARDUINO

Contrôler une valeur analogique manuellement

Le potentiomètre est une résistance variable à trois bornes. En tournant son axe, on modifie la tension lue sur la borne centrale. Arduino peut lire cette tension via une entrée analogique et l'utiliser pour contrôler divers systèmes.

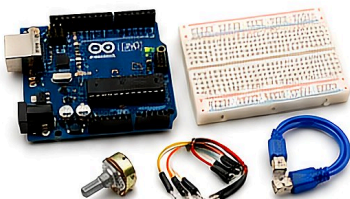


Le potentiomètre est très utile pour régler la vitesse d'un moteur, la luminosité d'une LED, le volume d'un buzzer, ou tout autre paramètre dans un projet électronique.



MATÉRIEL NÉCESSAIRE

- Arduino Uno 1
- Potentiomètre 10 kΩ 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1



Le potentiomètre utilisé est de type linéaire 10 kΩ (B10K).

SCHÉMA DE CÂBLAGE

Connectez le potentiomètre comme suit :



- Broche gauche → 5V
- Broche centrale → A0
- Broche droite → GND



Pourquoi ce câblage ?

Le potentiomètre forme un pont diviseur de tension réglable entre 0V et 5V. Arduino lit cette tension sur A0.

CODE ARDUINO



```
const int potPin = A0;
int valeurPot = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  valeurPot = analogRead(potPin);
  Serial.print("Valeur potentiomètre : ");
  Serial.println(valeurPot);
  delay(300);
}
```



Ouvrez le Moniteur Série à 9600 bauds pour voir les valeurs en temps réel.

EXPLICATION DU PROGRAMME

La fonction `analogRead(A0)` lit la tension sur la broche A0 et retourne une valeur entière entre 0 et 1023.

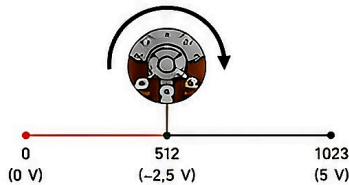
Interprétation des valeurs :

- 0 → potentiomètre tourné complètement vers GND
- 512 → position intermédiaire (environ 2.5 V)
- 1023 → potentiomètre tourné complètement vers 5V

Ces valeurs peuvent être utilisées pour piloter des actionneurs, ajuster des seuils ou afficher des mesures.

COMMENT FONCTIONNE UN POTENTIOMÈTRE

En tournant l'axe, la résistance entre la broche centrale et les extrémités change.



APPLICATIONS PRATIQUES



Réglage de vitesse d'un moteur DC avec PWM.



Contrôle de la luminosité d'une LED ou d'un écran.



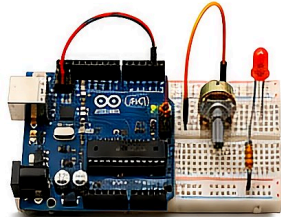
Ajustement du volume d'un buzzer ou d'un module audio.



Positionnement d'un servo moteur dans un projet robotique.

EXEMPLE PRATIQUE : CONTRÔLE DE LUMINOSITÉ D'UNE LED

- 1 Connectez le potentiomètre comme indiqué ci-dessus.
- 2 Connectez une LED à la broche 9 via une résistance de 220 Ω.
- 3 Téléversez le code ci-contre.
- 4 Tournez le potentiomètre et observez la variation de luminosité de la LED.



Code pour la LED

```
const int potPin = A0;
const int ledPin = 9;
int valeur = 0;

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  valeur = analogRead(potPin); // 0 à 1023
  int luminosite = map(valeur, 0, 1023, 0, 255);
  analogWrite(ledPin, luminosite); // PWM LED
  delay(10);
}
```



La fonction `map()` convertit la valeur lue (0–1023) en une sortie PWM (0–255).

CONSEILS D'AMÉLIORATION



Utilisez un potentiomètre de meilleure qualité pour plus de précision.



Ajoutez un condensateur de 100 nF entre 5V et GND pour filtrer le bruit.



Calibrez les valeurs min/max dans votre code si nécessaire.



ASTUCE INGÉNIEUR

Le potentiomètre est l'un des composants les plus simples et les plus puissants en électronique. Il permet à l'utilisateur d'interagir avec le système en temps réel de manière fluide et intuitive.



BON À SAVOIR

Certains potentiomètres sont linéaires (B10K), d'autres logarithmiques (A10K). Le linéaire est idéal pour les contrôles de position, le logarithmique pour le son.



À RETENIR



Le potentiomètre fournit une tension variable.



Arduino lit cette tension via `analogRead()`.



Les valeurs peuvent être utilisées pour contrôler des sorties analogiques ou logiques.

3

CHAPITRE 3

Arduino
et ESP32



Les capteurs sonores permettent de détecter le bruit et de mesurer son niveau en décibels (dB).

3.13

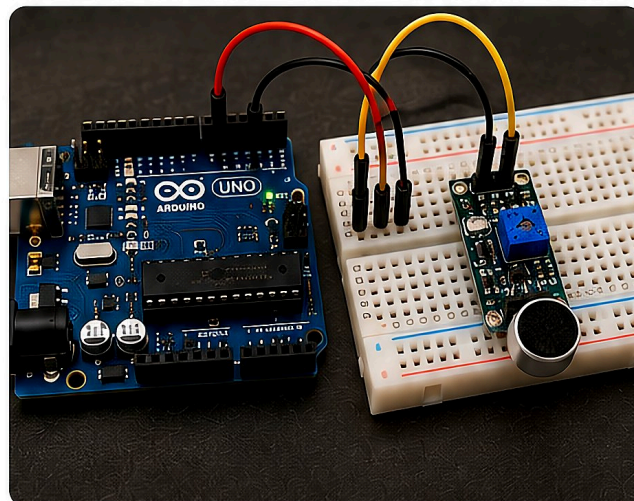
CAPTEUR SONORE (MICROPHONE) AVEC ARDUINO

Détecter et mesurer le son

Un capteur microphone convertit les ondes sonores en signaux électriques. Ces signaux peuvent être lus par Arduino et utilisés pour contrôler des actions, afficher des mesures ou déclencher des alertes.



Les capteurs sonores sont utilisés dans les systèmes d'alarme, les robots interactifs, les commandes vocales et les projets d'analyse audio.



MATÉRIEL NÉCESSAIRE

- Arduino Uno 1
- Module microphone (KY-037 ou similaire) 1
- Résistance 10 kΩ 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1

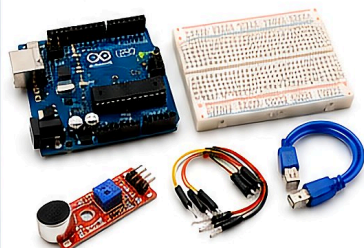
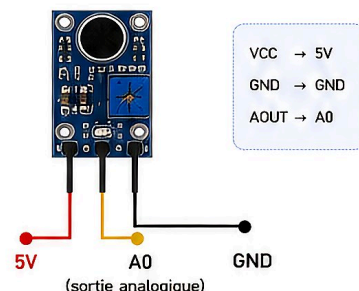


SCHÉMA DE CÂBLAGE

Branchez le module microphone comme suit :



Le potentiomètre du module permet d'ajuster la sensibilité du microphone.

CODE ARDUINO

```
const int micPin = A0;
int valeurSon = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  valeurSon = analogRead(micPin);
  Serial.print("Niveau sonore : ");
  Serial.println(valeurSon);
  delay(100);
}
```



Ouvrez le Moniteur Série à 9600 bauds pour voir les valeurs en temps réel.

EXPLICATION DU PROGRAMME

analogRead(A0) lit la tension générée par le microphone. Plus le son est fort, plus la valeur augmente.

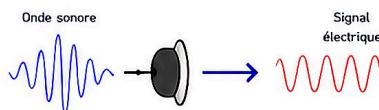
Plage des valeurs retournées par Arduino :

- 0 = silence complet
- 512 = bruit moyen
- 1023 = bruit très fort

Ces valeurs peuvent être utilisées pour déclencher des actions ou afficher le niveau sonore.

COMMENT FONCTIONNE UN MICROPHONE

Le microphone capte les vibrations de l'air et les convertit en signal analogique.



Ce signal varie en fonction de l'intensité et de la fréquence du son. Arduino le lit et le transforme en nombre.

APPLICATIONS POSSIBLES



Détection de claps pour allumer ou éteindre une LED.



Mesure du niveau sonore d'un environnement.



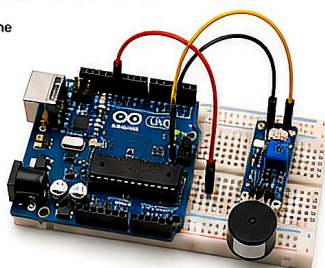
Systèmes d'alarme basés sur le bruit.



Robot qui réagit au son (parler, frapper, crier).

EXEMPLE PRATIQUE : ALARME SONORE

- 1 Branchez le module microphone comme indiqué ci-dessus.
- 2 Connectez un buzzer actif à la broche D8.
- 3 Téléversez le code ci-contre.
- 4 Faites du bruit (ex. : frapper dans vos mains).
- 5 Le buzzer retentit lorsque le bruit dépasse un seuil défini.



Code pour l'alarme sonore

```
const int micPin = A0;
const int buzzerPin = 8;
int seuil = 600; // seuil de détection
int valeurSon = 0;

void setup() {
  pinMode(buzzerPin, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  valeurSon = analogRead(micPin);
  Serial.println(valeurSon);

  if (valeurSon > seuil) {
    digitalWrite(buzzerPin, HIGH); // son ON
  } else {
    digitalWrite(buzzerPin, LOW); // son OFF
  }
  delay(100);
}
```



À RETENIR

- ✓ Le microphone convertit le son en signal analogique.
- ✓ Arduino lit ce signal à l'aide d'une entrée analogique.
- ✓ Les valeurs lues dépendent de l'intensité du son.
- ✓ Les capteurs sonores permettent de rendre les projets interactifs et intelligents.
- ✓ Ajustez toujours la sensibilité avec le potentiomètre pour obtenir de bonnes lectures.



IDÉES D'AMÉLIORATION

- Affichez le niveau sonore sur un écran LCD.
- Enregistrez les données sur une carte SD.
- Ajoutez des LED bargraph pour visualiser le niveau sonore.



BON À SAVOIR

Le microphone utilisé dans ces modules est de type électret. Il a besoin d'une tension de polarisation fournie par le module pour fonctionner correctement.



CONSEIL PRATIQUE

Évitez les sources de bruit parasite (moteurs, ventilateurs) lors des mesures pour obtenir des résultats fiables. Vous pouvez aussi lisser les valeurs avec une moyenne si nécessaire.

3

CHAPITRE 3

Arduino
et ESP32



Les ultrasons sont inaudibles pour l'oreille humaine (> 20 kHz) et peuvent être utilisés pour mesurer la distance avec précision.

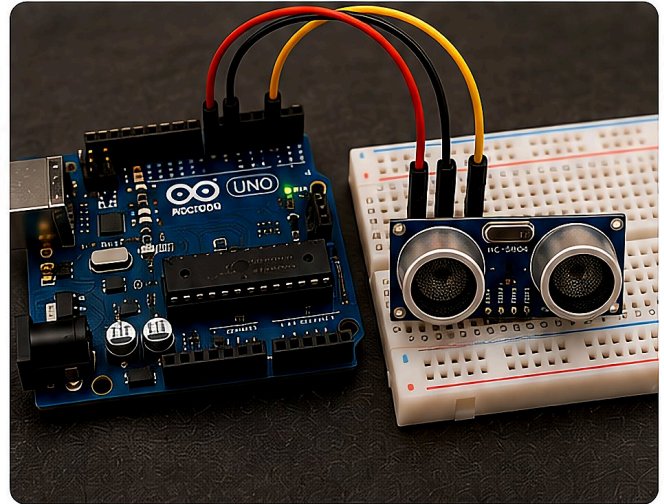
3.14

CAPTEUR DE DISTANCE ULTRASON HC-SR04 AVEC ARDUINO

Mesurer la distance sans contact

Le module ultrason HC-SR04 utilise des ondes sonores à haute fréquence pour mesurer la distance d'un objet. Il est largement utilisé dans les robots mobiles, les systèmes d'évitement d'obstacles et les projets d'automatisation.

i Le HC-SR04 émet une onde sonore et mesure le temps que prend l'écho pour revenir. Plus le temps est long, plus l'objet est loin.



MATÉRIEL NÉCESSAIRE

- Arduino Uno 1
- Module ultrason HC-SR04 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1

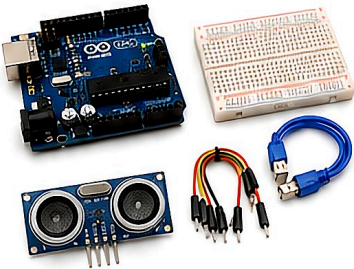
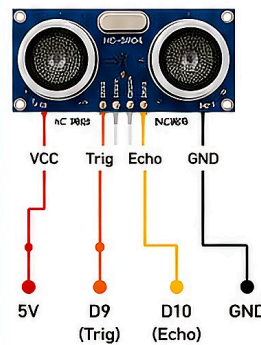


SCHÉMA DE CÂBLAGE

Branchez le module HC-SR04 comme suit :



- VCC → 5V
- Trig → D9
- Echo → D10
- GND → GND

i Le Trig envoie l'impulsion ultrason.
Le Echo reçoit l'écho et renvoie le temps de retour.

CODE ARDUINO

```
const int trigPin = 9;
const int echoPin = 10;
long duration;
float distanceCm;

void setup() {
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distanceCm = duration * 0.034 / 2;

  Serial.print("Distance : ");
  Serial.print(distanceCm);
  Serial.println(" cm");
  delay(500);
}
```



Ouvrez le Moniteur Série à 9600 bauds pour voir les mesures en temps réel.

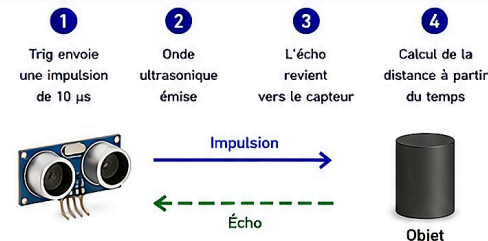
EXPLICATION DU PROGRAMME

Le module envoie une impulsion ultrasonique de 10 µs sur la broche Trig. Il attend ensuite l'écho sur la broche Echo. La fonction pulseIn() mesure la durée (en microsecondes) du temps de retour de l'écho. La distance est calculée avec la formule :

$$\text{Distance (cm)} = \text{durée (µs)} \times 0,034 / 2$$

0,034 cm/µs est la vitesse du son dans l'air (≈ 343 m/s).

COMMENT FONCTIONNE LE HC-SR04

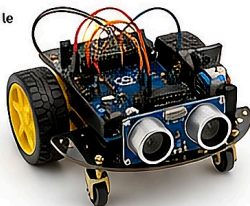


APPLICATIONS PRATIQUES

- Robots évitant les obstacles
- Systèmes de stationnement assisté
- Mesure de niveau de liquide dans des réservoirs
- Ouverture automatique de portes
- Projets de domotique et de sécurité

EXEMPLE PRATIQUE : ROBOT ÉVITEUR D'OBSTACLES

- 1 Assemblez le robot avec deux moteurs et le capteur HC-SR04 à l'avant.
- 2 Mesurez la distance en permanence.
- 3 Si la distance est inférieure à 20 cm, arrêtez le robot.
- 4 Faites reculer légèrement.
- 5 Tournez à gauche ou à droite pour contourner l'obstacle.



DISTANCE VS ACTION

Distance mesurée	Action du robot
> 30 cm	Avancer
20 – 30 cm	Ralentir
10 – 20 cm	Préparer à éviter
< 10 cm	Reculer et tourner



À RETENIR

- ✓ Le HC-SR04 mesure la distance à l'aide d'ultrasons.
- ✓ Il utilise deux broches : Trig (émission) et Echo (réception).
- ✓ La distance dépend du temps de retour de l'écho.
- ✓ Idéal pour les robots, la sécurité et les systèmes intelligents.



IDÉES D'AMÉLIORATION

- Affichez la distance sur un écran LCD ou OLED.
- Utilisez plusieurs capteurs pour une détection 360°.
- Combinez avec des moteurs et un contrôleur de navigation.



BON À SAVOIR

Le HC-SR04 a une portée de 2 cm à 400 cm avec un angle de détection d'environ 15°. Les surfaces mousses ou inclinées peuvent réduire la précision.



CONSEIL PRATIQUE

Évitez d'utiliser le capteur dans des environnements très bruyants (ultrasons parasites). Testez toujours plusieurs fois pour obtenir des mesures fiables.

3

CHAPITRE 3

Arduino
et ESP32



Les capteurs infrarouges utilisent la lumière IR pour détecter les objets sans contact. Parfaits pour l'automatisation et la robotique.

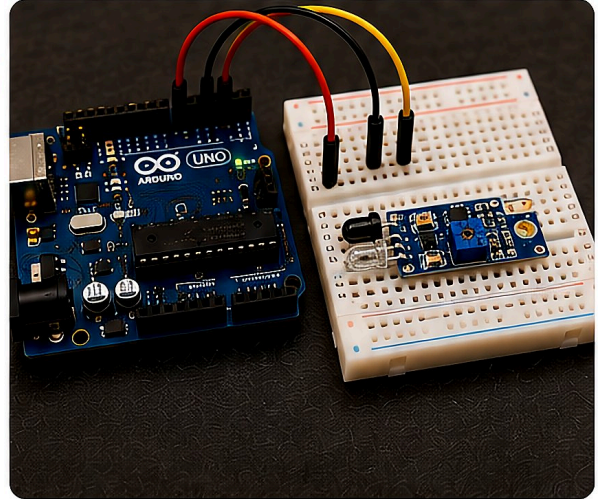
3.15

CAPTEUR INFRAROUGE (IR) DE PROXIMITÉ AVEC ARDUINO

Détecter la présence d'un objet

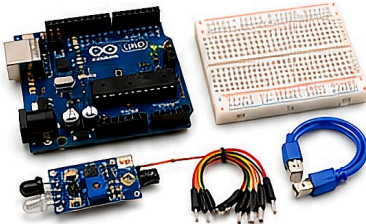
Le capteur infrarouge de proximité détecte la présence d'un objet devant lui à l'aide d'une LED infrarouge (émission) et d'un phototransistor (réception). Il fournit une sortie numérique : HIGH (objet détecté) ou LOW (aucun objet).

i Ce capteur est très utilisé dans les robots suiveurs de ligne, les systèmes d'évitement d'obstacles et les compteurs d'objets.



MATÉRIEL NÉCESSAIRE

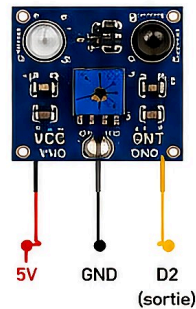
- Arduino Uno 1
- Module capteur IR de proximité 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1



i Le module possède un potentiomètre pour ajuster la distance de détection.

SCHÉMA DE CÂBLAGE

Connectez le module IR comme suit :



VCC → 5V
GND → GND
OUT → D2

i La sortie OUT fournit un signal numérique. LOW = aucun objet détecté. HIGH = objet détecté.

CODE ARDUINO



```
const int capteurIR = 2; // Broche de sortie du module IR
int etat = 0;

void setup() {
  pinMode(capteurIR, INPUT);
  Serial.begin(9600);
}

void loop() {
  etat = digitalRead(capteurIR);

  if (etat == LOW) {
    Serial.println("Aucun objet détecté");
  } else {
    Serial.println("Objet détecté !");
  }

  delay(200);
}
```



Ouvrez le Moniteur Série à 9600 bauds pour voir les résultats en temps réel.

EXPLICATION DU PROGRAMME

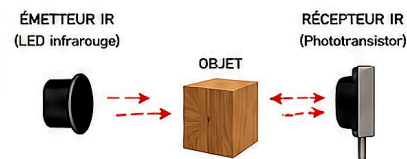
Le capteur IR envoie un faisceau infrarouge. Lorsqu'un objet se trouve dans la zone, la lumière IR est réfléchiée et reçue par le phototransistor.

La broche OUT passe alors à HIGH (5V). Sinon, elle reste à LOW (0V).

- LOW → Aucun objet détecté
- HIGH → Objet détecté

Ces informations sont lues par Arduino et affichées dans le Moniteur Série.

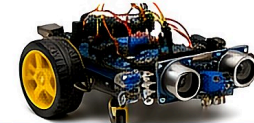
COMMENT FONCTIONNE LE CAPTEUR IR



La lumière IR émise se reflète sur l'objet et revient vers le récepteur. Plus l'objet est proche, plus la détection est fiable.

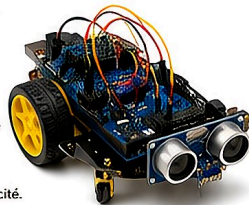
APPLICATIONS PRATIQUES

- Robots évitant les obstacles.
- Systèmes de sécurité et d'alarme.
- Compteurs d'objets sur une chaîne.
- Détection de passage ou de présence.
- Ports automatiques.



EXEMPLE PRATIQUE : ROBOT ÉVITEUR D'OBSTACLES

- 1 Assemblez le robot avec deux moteurs.
- 2 Fixez le capteur IR à l'avant du robot.
- 3 Branchez le capteur comme indiqué ci-dessus.
- 4 Si un objet est détecté, le robot s'arrête ou change de direction.
- 5 Vous pouvez combiner ce capteur avec un servo ou plusieurs pour plus d'efficacité.



DISTANCE DE DÉTECTION (APPROX.)

Réglage du potentiomètre	Distance de détection
Minimum (sensibilité faible)	2 – 10 cm
Moyen	5 – 30 cm
Maximum (sensibilité élevée)	10 – 80 cm



La distance peut varier selon la couleur, la taille et la surface de l'objet.



À RETENIR

- ✓ Le capteur IR détecte la présence d'un objet grâce à la lumière infrarouge.
- ✓ Sortie numérique : LOW ou HIGH.
- ✓ Facile à utiliser et à intégrer avec Arduino.
- ✓ Idéal pour la robotique et l'automatisation.
- ✓ Ajustez toujours le potentiomètre pour une détection optimale.



IDÉES D'AMÉLIORATION

- Ajoutez un buzzer pour alerter quand un objet est détecté.
- Utilisez plusieurs capteurs IR pour une détection à 360°.
- Combinez-le avec un écran LCD pour afficher la distance estimée.



BON À SAVOIR

Les capteurs IR sont sensibles à la lumière ambiante. Évitez une exposition directe au soleil ou à des lampes très puissantes. Testez toujours votre système dans différentes conditions.



CONSEIL PRATIQUE

Nettoyez régulièrement la lentille du capteur (LED IR et récepteur) pour éviter la poussière qui peut réduire la portée de détection.

3

CHAPITRE 3

Arduino
et ESP32



Les servomoteurs peuvent se positionner avec précision sur un angle allant généralement de 0° à 180°.

3.16

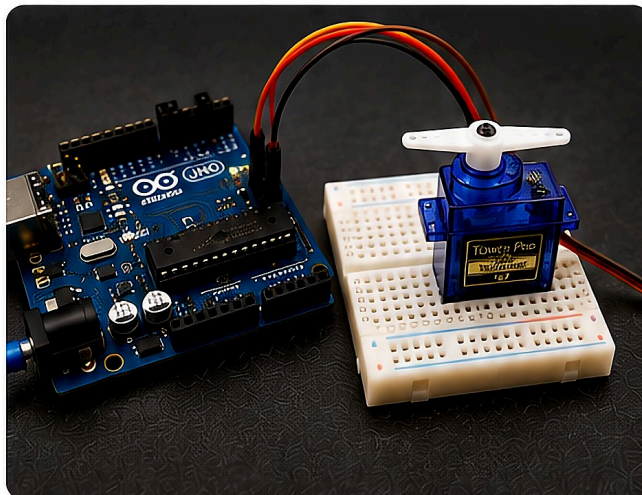
SERVOMOTEUR AVEC ARDUINO

Contrôler la position avec précision

Le servomoteur est un actionneur capable de se positionner à un angle précis. Il est largement utilisé dans la robotique, les bras mécaniques, les véhicules télécommandés et de nombreux projets d'automatisation.



Un servomoteur reçoit un signal PWM (Pulse Width Modulation) qui détermine l'angle de sa position.



MATÉRIEL NÉCESSAIRE

- Arduino Uno 1
- Servomoteur SG90 1
- Fils de connexion Plusieurs
- Breadboard 1
- Câble USB 1

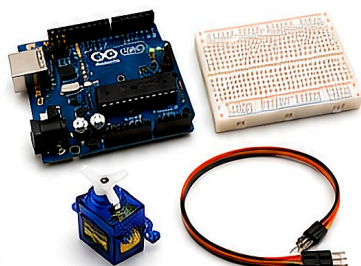
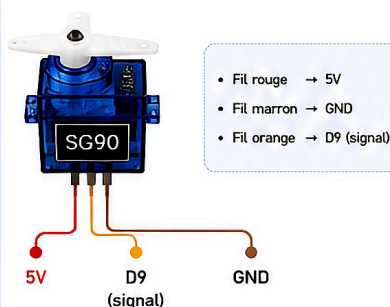


SCHÉMA DE CÂBLAGE

Branchez le servomoteur comme suit :



Le signal PWM est envoyé via la broche D9.

CODE ARDUINO



```
#include <Servo.h>

Servo monServo;

void setup() {
  monServo.attach(9); // Attache le servo à D9
}

void loop() {
  monServo.write(0); // Position 0°
  delay(1000);

  monServo.write(90); // Position 90°
  delay(1000);

  monServo.write(180); // Position 180°
  delay(1000);

  monServo.write(90); // Retour au centre
  delay(1000);
}
```



Ouvrez le Moniteur Série à 9600 bauds (optionnel pour afficher des messages).

EXPLICATION DU PROGRAMME

La bibliothèque Servo.h simplifie le contrôle du servomoteur.

- attach(9) → définit la broche de signal (D9).
- write(angle) → déplace le servo à l'angle spécifié.

La fonction write() accepte des valeurs de :

- 0° → position minimale
- 90° → position centrale
- 180° → position maximale

Le délai de 1000 ms permet de voir le mouvement à chaque étape.

COMMENT FONCTIONNE UN SERVOMOTEUR

Le servomoteur contient un moteur DC, un engrenage de réduction et un potentiomètre.



Le potentiomètre mesure l'angle actuel et le compare à la commande reçue. Le servo ajuste automatiquement sa position.

APPLICATIONS COURANTES



Bras robotiques et manipulateurs



Voitures et avions RC



Pan-tilt pour caméras



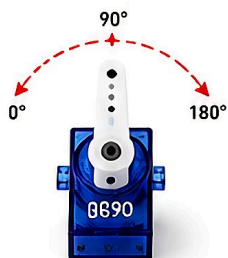
Ouverture de portes et trappes



Projets éducatifs et maquettes

EXEMPLE PRATIQUE : BALAYAGE 0° À 180°

- 1 Assurez-vous que le câblage est correct.
- 2 Téléversez le code ci-contre.
- 3 Le servo se déplacera de 0° à 180° puis reviendra au centre.
- 4 Changez les valeurs dans write() pour contrôler d'autres positions.



Code : balayage continu

```
#include <Servo.h>
Servo monServo;

void setup() {
  monServo.attach(9);
}

void loop() {
  for (int angle = 0; angle <= 180; angle += 1) {
    monServo.write(angle);
    delay(15);
  }
  for (int angle = 180; angle >= 0; angle -= 1) {
    monServo.write(angle);
    delay(15);
  }
}
```



À RETENIR

- ✓ Le servomoteur se déplace à un angle précis.
- ✓ Il est contrôlé par un signal PWM via une broche numérique.
- ✓ Les angles vont généralement de 0° à 180°.
- ✓ Idéal pour les mouvements contrôlés et les projets robotiques.
- ✓ La bibliothèque Servo.h facilite son utilisation.



IDÉES D'AMÉLIORATION

- Contrôlez le servo avec un potentiomètre.
- Utilisez plusieurs servos pour créer un bras robotisé.
- Combinez avec des capteurs (distance, IR) pour des systèmes intelligents.



BON À SAVOIR

Certains servomoteurs peuvent tourner en continu (360°) au lieu d'avoir un angle limité. Dans ce cas, la commande write(90) correspond généralement à l'arrêt.



CONSEIL PRATIQUE

Alimentez les servomoteurs avec une source 5V stable capable de fournir suffisamment de courant (1A ou plus selon le nombre de servos utilisés).

3

CHAPITRE 3

Arduino
et ESP32



Les moteurs pas à pas offrent un contrôle précis de la position, idéal pour les systèmes qui nécessitent des mouvements répétitifs et fiables.

3.17

MOTEUR PAS À PAS (STEPPER) AVEC ARDUINO

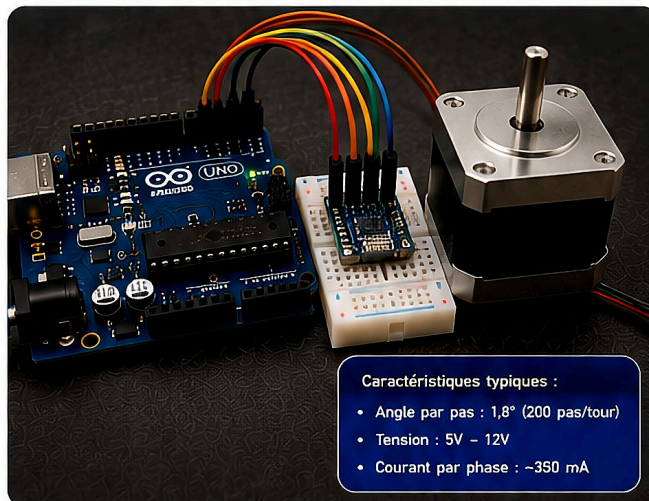
Des mouvements précis, pas par pas

Le moteur pas à pas convertit des impulsions électriques en déplacements angulaires discrets. Chaque impulsion fait tourner le moteur d'un petit angle appelé "pas".

Il est largement utilisé dans l'impression 3D, les machines CNC, les scanners, les bras robotiques et les systèmes de positionnement.



Un moteur pas à pas ne tourne pas continuellement comme un moteur DC. Il avance par étapes contrôlées, offrant une grande précision sans capteur de retour.

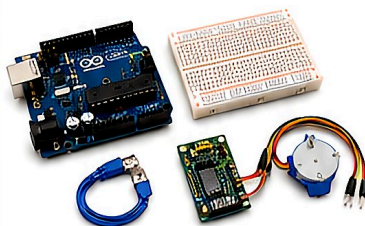


Caractéristiques typiques :

- Angle par pas : 1,8° (200 pas/tour)
- Tension : 5V – 12V
- Courant par phase : ~350 mA

MATÉRIEL NÉCESSAIRE

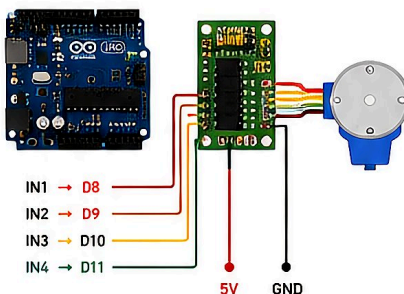
- Arduino Uno 1
- Moteur pas à pas 28BYJ-48 1
- Module driver ULN2003 1
- Breadboard 1
- Fils de connexion Plusieurs
- Alimentation externe (recommandée) 1
- Câble USB 1



Le module ULN2003 simplifie le câblage et permet d'alimenter le moteur facilement.

SCHÉMA DE CÂBLAGE

Branchez le moteur pas à pas 28BYJ-48 avec le module ULN2003 :



Le moteur reçoit l'alimentation via le module ULN2003. Ne connectez pas 5V d'Arduino directement au moteur s'il utilise une alimentation séparée.

CODE ARDUINO (LIBRAIRIE STEPPER)

```
#include <Stepper.h>

const int stepsPerRevolution = 2048; // 28BYJ-48
Stepper myStepper(stepsPerRevolution, 8, 10, 9, 11);

void setup() {
  myStepper.setSpeed(10); // 10 tr/min
}

void loop() {
  // Rotation dans le sens horaire
  myStepper.step(stepsPerRevolution); // 1 tour
  delay(500);

  // Rotation dans le sens antihoraire
  myStepper.step(-stepsPerRevolution); // 1 tour inverse
  delay(500);
}
```



Ouvrez le Moniteur Série à 9600 bauds (optionnel pour afficher des messages).

EXPLICATION DU PROGRAMME

La bibliothèque Stepper.h permet de contrôler facilement le moteur pas à pas.

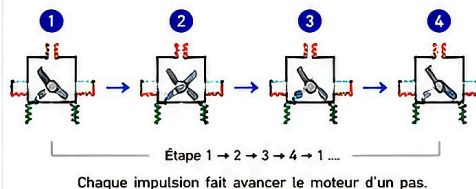
Fonction step(n) :

- Si n est positif → le moteur tourne dans un sens.
- Si n est négatif → le moteur tourne dans l'autre sens.
- stepsPerRevolution : nombre de pas pour 1 tour complet.

Pour le 28BYJ-48 : 1 tour = 2048 pas (360°)
Donc : 1 pas = 360 / 2048 ≈ 0,1758°

COMMENT FONCTIONNE UN MOTEUR PAS À PAS

Le moteur est composé de 4 bobines (phases). Le driver active les bobines dans un ordre précis pour créer un champ magnétique tournant qui fait avancer le rotor.



APPLICATIONS COURANTES



Imprimantes 3D et CNC



Systèmes de positionnement de caméras



Robots et bras mécaniques



Machines de découpe et gravure



Automatisation industrielle



Distributeurs automatiques

EXEMPLE PRATIQUE : CONTRÔLE DE L'ANGLE

- 1 Modifiez le code pour tourner le moteur d'un angle spécifique.
- 2 90° = 512 pas (90/360 × 2048 = 512)
- 3 180° = 1024 pas
- 4 45° = 256 pas
- 5 Testez différents angles !

Code : Angle personnalisé

```
const int stepsPerRevolution = 2048;
Stepper myStepper(stepsPerRevolution, 8, 10, 9, 11);

void setup() {
  myStepper.setSpeed(10); // 10 tr/min
}

void loop() {
  myStepper.step(512); // Tourne de 90°
  delay(1000);
  myStepper.step(-1024); // Tourne de -180°
  delay(1000);
  myStepper.step(256); // Tourne de 45°
  delay(1000);
}
```



À RETENIR

- ✓ Le moteur pas à pas avance par étapes précises.
- ✓ Le module ULN2003 facilite le pilotage du 28BYJ-48.
- ✓ Le nombre de pas détermine la résolution du mouvement.
- ✓ Idéal pour les projets qui nécessitent une position exacte et répétable.
- ✓ Vous pouvez combiner le moteur avec des capteurs ou un écran LCD pour des systèmes avancés.



IDÉES D'AMÉLIORATION

- Utilisez un afficheur LCD pour montrer l'angle actuel.
- Ajoutez des boutons pour contrôler l'angle.
- Utilisez un encodeur rotatif pour un réglage précis.



BON À SAVOIR

Pour plus de couple, utilisez une alimentation externe (5V 1A ou plus). Ne dépassez pas le courant maximal du moteur pour éviter la surchauffe.



CONSEIL PRATIQUE

Testez d'abord le moteur lentement (vitesse faible) pour vous assurer que le câblage et le code fonctionnent correctement.

3

CHAPITRE 3

Arduino
et ESP32



Les écrans LCD affichent des informations utiles (par exemple des mesures, des états ou des messages) dans vos projets.

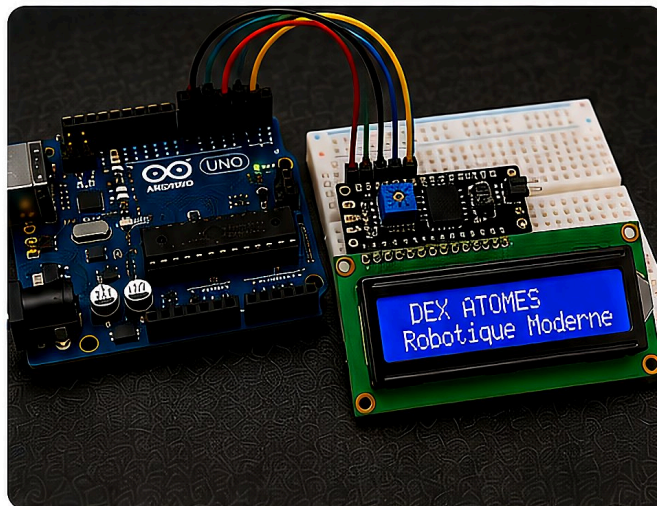
3.18

AFFICHEUR LCD 16x2 AVEC ARDUINO

Afficher des informations facilement

L'écran LCD 16x2 permet d'afficher jusqu'à 16 caractères par ligne sur 2 lignes. Il est parfait pour les interfaces utilisateur, les mesures de capteurs, les menus et l'état des systèmes.

i Nous utiliserons un module LCD avec interface I2C (LCM1602 I2C). Il ne nécessite que 4 connexions avec Arduino (SDA, SCL, 5V, GND).



MATÉRIEL NÉCESSAIRE

- Arduino Uno 1
- Écran LCD 16x2 avec I2C 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1

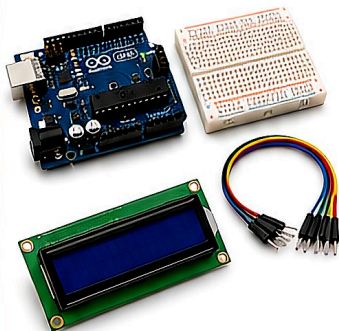
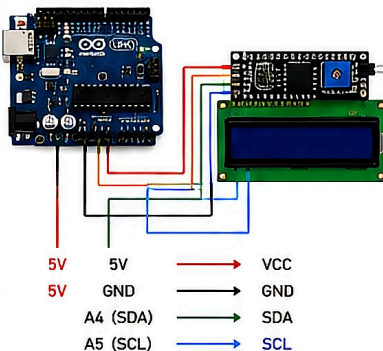


SCHÉMA DE CÂBLAGE (I2C)

Branchez l'écran LCD comme suit :



i Sur Arduino Uno : A4 = SDA et A5 = SCL.

CODE ARDUINO (LIBRAIRIE LIQUIDCRYSTAL_I2C)



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

LiquidCrystal_I2C lcd(0x27, 16, 2); // Adresse I2C (0x27 ou 0x3F)

void setup() {
  lcd.init();           // Initialise l'écran
  lcd.backlight();      // Allume le rétroéclairage
  lcd.setCursor(0, 0);
  lcd.print("DEX ATOMES");
  lcd.setCursor(0, 1);
  lcd.print("Robotique Moderne");
}

void loop() {
  lcd.setCursor(0, 0);
  lcd.print("Temp: 25.3C "); // Affiche et efface la fin
  delay(2000);
  lcd.setCursor(0, 1);
  lcd.print("Humidite: 60% ");
  delay(2000);
}
```

i Ouvrez le Moniteur Série à 9600 bauds (optionnel pour afficher des messages).

EXPLICATION DU PROGRAMME

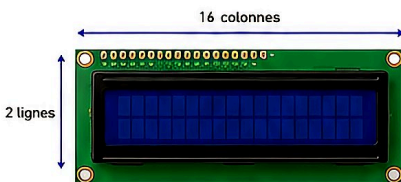
La bibliothèque LiquidCrystal_I2C permet de communiquer avec l'écran via le protocole I2C.

- init() → Initialise l'écran LCD
- backlight() → Active le rétroéclairage
- setCursor(col, row) → Définit la position du curseur
- print() → Affiche le texte

Les coordonnées commencent à (0,0) en haut à gauche.

- row = 0 → première ligne
- row = 1 → deuxième ligne

STRUCTURE DE L'AFFICHEUR LCD 16x2



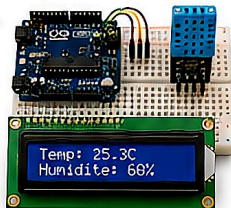
- Chaque ligne contient 16 caractères.
- Caractères disponibles : lettres, chiffres, symboles.
- Contraste réglable avec le potentiomètre du module I2C.

APPLICATIONS POSSIBLES

- Affichage des valeurs de capteurs (température, distance, etc.)
- Menus et réglages dans les projets
- Affichage de messages d'état ou d'alertes
- Interfaces utilisateur pour robots
- Affichage de l'heure, compteurs, scores

EXEMPLE PRATIQUE : STATION MÉTÉO SIMPLE

- 1 Connectez un capteur de température (DHT11/DHT22) à Arduino.
- 2 Affichez la température et l'humidité sur l'écran LCD.
- 3 Mettez à jour les valeurs toutes les 2 secondes.



Extrait de code :

```
lcd.setCursor(0, 0);
lcd.print("Temp: ");
lcd.print(temperature);
lcd.print(" C ");

lcd.setCursor(0, 1);
lcd.print("Humidite: ");
lcd.print("Humidite");
lcd.print(" % ");
```



À RETENIR

- ✓ L'écran LCD 16x2 permet d'afficher des textes sur 2 lignes de 16 caractères.
- ✓ Le module I2C simplifie le câblage (4 fils seulement).
- ✓ Les fonctions principales : init(), backlight(), setCursor(), print().
- ✓ Idéal pour les interfaces, les mesures et les messages dans vos projets.



IDÉES D'AMÉLIORATION

- Ajustez le contraste avec le potentiomètre si le texte est pâle ou trop foncé.
- Essayez d'autres adresses I2C (0x27 ou 0x3F) si l'écran ne s'affiche pas.
- Ajoutez des animations ou un défilement de texte.



BON À SAVOIR

Plusieurs périphériques I2C peuvent partager les mêmes broches SDA et SCL, ce qui permet de connecter plusieurs capteurs et écrans simultanément.



CONSEIL PRATIQUE

Utilisez un écran LCD rétroéclairé pour une meilleure lisibilité. Dans les projets sur batterie, éteignez le rétroéclairage quand il n'est pas nécessaire pour économiser de l'énergie.

3

CHAPITRE 3

Arduino
et ESP32



Les boutons poussoirs permettent à l'utilisateur d'interagir avec le système et déclencher des actions.

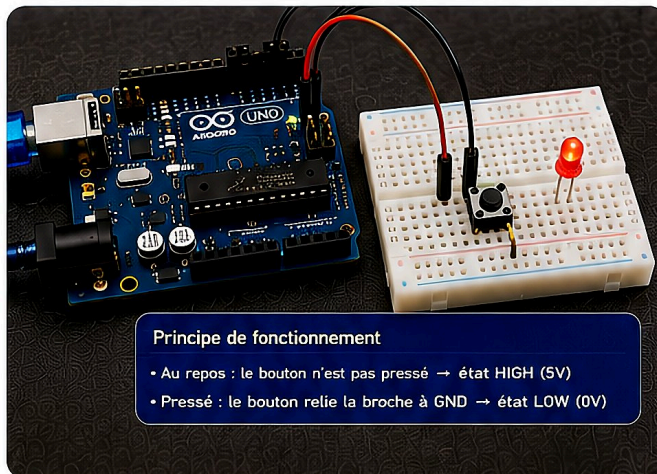
3.19

BOUTON POUSSOIR AVEC ARDUINO

Détecter une pression et exécuter une action

Le bouton poussoir est un composant simple qui permet de commander un circuit lorsqu'il est pressé. Il est largement utilisé dans les interfaces utilisateur, les robots, les systèmes de contrôle et les projets interactifs.

i Nous utiliserons la résistance de tirage interne (INPUT_PULLUP) pour éviter les résistances externes et simplifier le câblage.

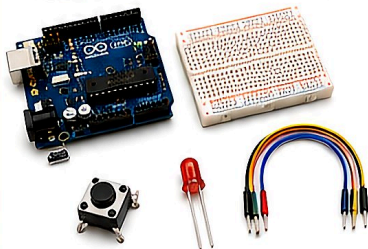


Principe de fonctionnement

- Au repos : le bouton n'est pas pressé → état HIGH (5V)
- Pressé : le bouton relie la broche à GND → état LOW (0V)

MATÉRIEL NÉCESSAIRE

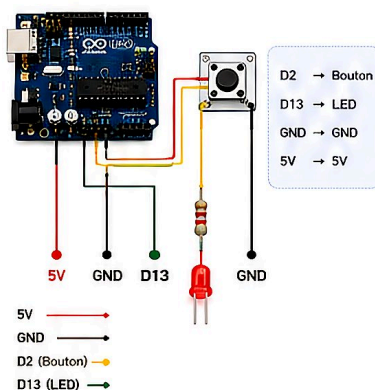
- Arduino Uno 1
- Bouton poussoir (tactile) 1
- LED (5 mm) 1
- Résistance 220 Ω 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1



i Nous utiliserons la résistance de tirage interne (INPUT_PULLUP) pour simplifier le montage.

SCHEMA DE CÂBLAGE

Branchez le bouton poussoir et la LED comme suit :



i Le bouton relie la broche D2 à la masse (GND) lorsqu'il est pressé.

CODE ARDUINO

```
const int boutonPin = 2; // Broche du bouton
const int ledPin = 13; // Broche de la LED

void setup() {
  pinMode(boutonPin, INPUT_PULLUP); // Activation de la // résistance interne
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW); // Éteint la LED au départ
  Serial.begin(9600);
}

void loop() {
  int etat = digitalRead(boutonPin);

  if (etat == LOW) { // Bouton pressé
    digitalWrite(ledPin, HIGH); // Allume la LED
    Serial.println("Bouton pressé !");
    delay(200); // Anti-rebond simple
  } else {
    digitalWrite(ledPin, LOW); // Bouton relâché
  }
}
```

Ouvrez le Moniteur Série à 9600 bauds pour voir les messages lorsque le bouton est pressé.

EXPLICATION DU PROGRAMME

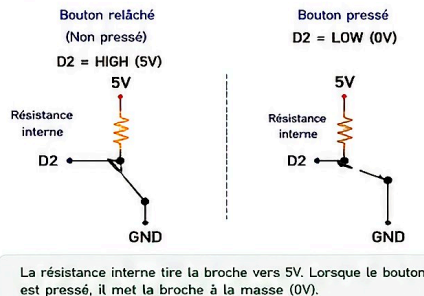
Le bouton est connecté à D2 et à GND. Grâce à la résistance de tirage interne (INPUT_PULLUP) :

- Bouton relâché → état HIGH (1)
- Bouton pressé → état LOW (0)

Lorsque le bouton est pressé, la LED connectée à D13 s'allume et un message s'affiche dans le Moniteur Série.

Fonction utilisée	Rôle
pinMode(..., INPUT_PULLUP)	Active la résistance interne
digitalRead(...)	Lit l'état du bouton
digitalWrite(...)	Allume ou éteint la LED
delay()	Évite les lectures multiples dues à l'effet de rebond

COMMENT FONCTIONNE LE BOUTON POUSSOIR

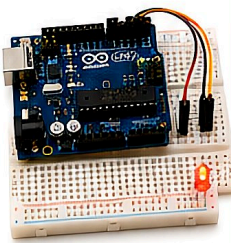


APPLICATIONS PRATIQUES

- Contrôle de menus sur écrans LCD ou OLED
- Démarrage / arrêt d'un système
- Navigation dans les interfaces utilisateur
- Réinitialisation (reset) manuelle
- Jeux électroniques et projets interactifs
- Commandes dans les robots

EXEMPLE PRATIQUE : COMPTEUR DE PRESSION

- 1 À chaque pression sur le bouton, le compteur augmente.
- 2 Le nombre de pressions s'affiche sur le Moniteur Série.
- 3 La LED clignote à chaque pression.
- 4 Maintenez le bouton appuyé 2 secondes pour réinitialiser le compteur.



Code : Compteur de pressions

```
const int boutonPin = 2;
const int ledPin = 13;
int compteur = 0;
unsigned long pressStart = 0;
bool resetMode = false;

void setup() {
  pinMode(boutonPin, INPUT_PULLUP);
  pinMode(ledPin, OUTPUT);
  Serial.begin(9600);
  Serial.println("Compteur de pressions");
}

void loop() {
  int etat = digitalRead(boutonPin);
  if (etat == LOW) {
    if (!resetMode) {
      compteur++;
      digitalWrite(ledPin, HIGH);
      Serial.println("Pressions : ");
      Serial.println(compteur);
      delay(200);
    }
    if (millis() - pressStart > 2000) resetMode = true;
  } else {
    digitalWrite(ledPin, LOW);
    if (resetMode) {
      Serial.println("Compteur réinitialisé");
      resetMode = false;
      pressStart = millis();
    }
  }
}
```



À RETENIR

- ✓ Le bouton poussoir est simple et très utile.
- ✓ Utilisez INPUT_PULLUP pour éviter les résistances externes.
- ✓ Bouton relâché = HIGH. Bouton pressé = LOW.
- ✓ Pensez à gérer l'effet de rebond (delay ou méthodes avancées).
- ✓ Idéal pour les interfaces utilisateur et les projets interactifs.



ATTENTION

Si vous utilisez INPUT sans résistance interne, vous devez ajouter une résistance de tirage (10 kΩ) vers 5V pour éviter des lectures aléatoires.



IDÉES D'AMÉLIORATION

- Utilisez plusieurs boutons pour différentes actions.
- Combinez avec un encodeur rotatif pour naviguer dans les menus.
- Ajoutez un écran LCD pour afficher le compteur.



BON À SAVOIR

L'effet de rebond est un phénomène mécanique qui provoque plusieurs lectures lors d'une seule pression. Un petit délai (200 ms) ou une détection avancée permet de le réduire.



CONSEIL PRATIQUE

Testez toujours votre bouton avec le Moniteur Série avant de l'intégrer dans un projet plus complexe.

3

CHAPITRE 3

Arduino
et ESP32



Le buzzer produit un son lorsqu'il reçoit un signal électrique. Il est utilisé pour les alarmes, les notifications sonores et les retours utilisateur.

3.20

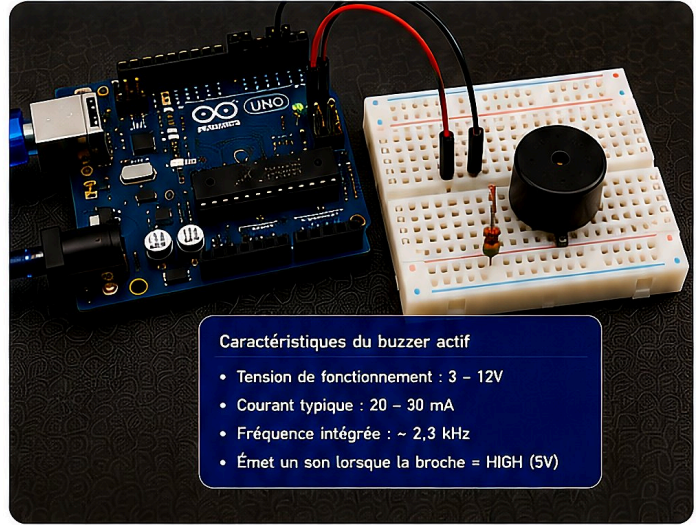
BUZZER ACTIF AVEC ARDUINO

Générer des sons et des alertes

Le buzzer actif intègre un oscillateur interne. Il émet un son lorsque l'on applique un signal HIGH (5V). Il existe en deux types : actif (avec oscillateur) et passif (sans oscillateur). Dans cette page, nous utilisons un buzzer actif pour simplifier le câblage et le code.



Le buzzer actif ne nécessite qu'une seule broche numérique pour fonctionner.

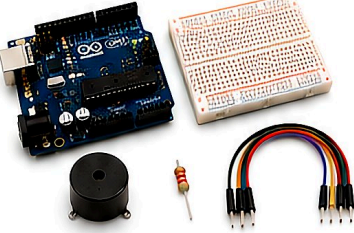


Caractéristiques du buzzer actif

- Tension de fonctionnement : 3 – 12V
- Courant typique : 20 – 30 mA
- Fréquence intégrée : ~ 2,3 kHz
- Émet un son lorsque la broche = HIGH (5V)

MATÉRIEL NÉCESSAIRE

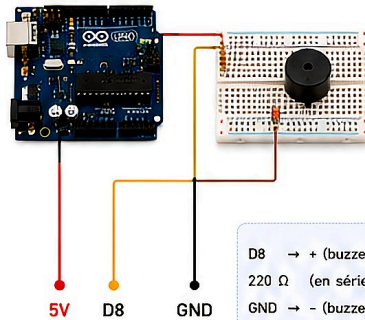
- Arduino Uno 1
- Buzzer actif 5V 1
- Résistance 220 Ω 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1



La résistance limite le courant pour protéger le buzzer et la broche de l'Arduino.

SCHEMA DE CÂBLAGE

Branchez le buzzer actif comme suit :



Le buzzer actif possède 2 broches : + (positif) et - (négatif).

CODE ARDUINO



```
const int buzzerPin = 8; // Broche du buzzer

void setup() {
  pinMode(buzzerPin, OUTPUT);
}

void loop() {
  digitalWrite(buzzerPin, HIGH); // Son ON
  delay(1000); // Attend 1 seconde

  digitalWrite(buzzerPin, LOW); // Son OFF
  delay(1000); // Attend 1 seconde
}
```



Ouvrez le Moniteur Série à 9600 bauds (optionnel, ce code n'affiche rien).

EXPLICATION DU PROGRAMME

Le programme active la broche D8 (buzzerPin) pendant 1 seconde puis la désactive pendant 1 seconde, en boucle infinie.

Instruction	Rôle
<code>pinMode(buzzerPin, OUTPUT)</code>	Définit la broche comme sortie
<code>digitalWrite(HIGH)</code>	Envoie 5V au buzzer (son ON)
<code>digitalWrite(LOW)</code>	Envoie 0V au buzzer (son OFF)
<code>delay(1000)</code>	Attend 1000 millisecondes (1s)



Pour changer la durée du son, modifiez la valeur du `delay()`. Pour un son continu, gardez la broche à HIGH.

COMMENT FONCTIONNE LE BUZZER ACTIF

Le buzzer actif contient un circuit oscillateur interne. Quand il reçoit 5V (HIGH), il génère automatiquement un son à sa fréquence interne.



Pas besoin de générer une fréquence avec `tone()`. Le signal HIGH suffit pour produire le son.

APPLICATIONS COURANTES



Alarmes et systèmes de sécurité



Notifications sonores dans les interfaces



Jeux électroniques et jouets



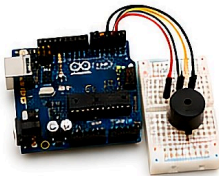
Robots et projets interactifs



Systèmes d'alerte et d'avertissement

EXEMPLE PRATIQUE : ALARME SIMPLE

- 1 Allumez le buzzer pendant 3 sons courts.
- 2 Chaque son dure 200 ms.
- 3 Pause de 150 ms entre chaque son.
- 4 Répétez la séquence toutes les 2 secondes.



Code : 3 sons courts

```
const int buzzer = 8;

void setup() { pinMode(buzzer, OUTPUT); }

void loop() {
  for (int i = 0; i < 3; i++) { // 3 sons courts
    digitalWrite(buzzer, HIGH); // Son ON
    delay(200); // 200 ms
    digitalWrite(buzzer, LOW); // Son OFF
    delay(150); // Pause 150 ms
  }
  delay(2000); // Attente 2 s
}
```



À RETENIR

- ✓ Le buzzer actif émet un son quand la broche est à HIGH (5V).
- ✓ Une seule broche numérique suffit pour le faire fonctionner.
- ✓ Utilisez une résistance (220 Ω) pour limiter le courant.
- ✓ Le buzzer est utile pour les alertes et le retour utilisateur.



IDÉES D'AMÉLIORATION

- Créez différentes mélodies avec des sons de durées variées.
- Ajoutez un bouton pour activer/désactiver le son.
- Combinez avec des capteurs pour des alertes intelligentes.



BON À SAVOIR

Pour le buzzer passif, vous devez utiliser la fonction `tone(pin, fréquence)` pour générer le son à une fréquence spécifique.



CONSEIL PRATIQUE

Si le son est trop fort ou désagréable, vous pouvez ajouter une petite résistance (100 à 220 Ω) ou utiliser un transistor pour contrôler le courant.

3

CHAPITRE 3

Arduino
et ESP32



Les capteurs de pression permettent de mesurer une force ou un poids. Ils sont essentiels dans la robotique et l'automatisation.

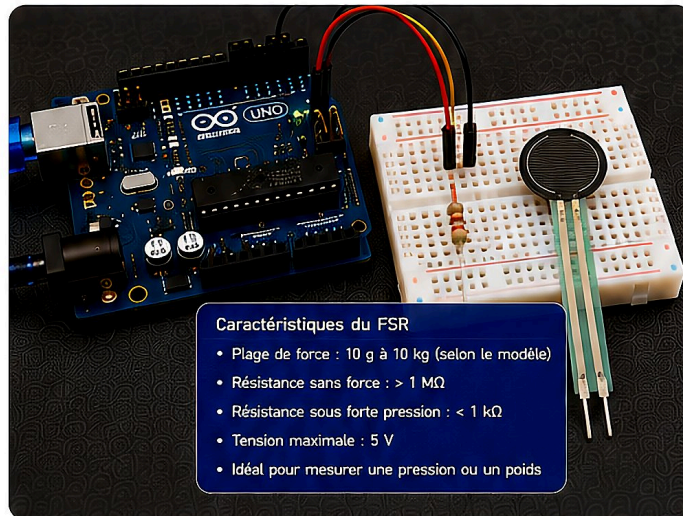
3.21

CAPTEUR DE PRESSION (FSR) AVEC ARDUINO

Mesurer une force ou un appui

Le capteur de force FSR (Force Sensitive Resistor) change sa résistance en fonction de la pression appliquée à sa surface. Plus la force est grande, plus la résistance diminue. Nous lirons cette variation avec Arduino pour afficher la valeur et la force estimée.

i Le FSR n'a pas de polarité. Il se comporte comme une résistance variable.

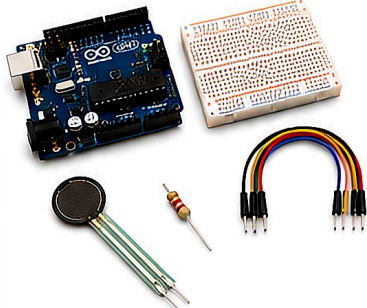


Caractéristiques du FSR

- Plage de force : 10 g à 10 kg (selon le modèle)
- Résistance sans force : > 1 MΩ
- Résistance sous forte pression : < 1 kΩ
- Tension maximale : 5 V
- Idéal pour mesurer une pression ou un poids

MATÉRIEL NÉCESSAIRE

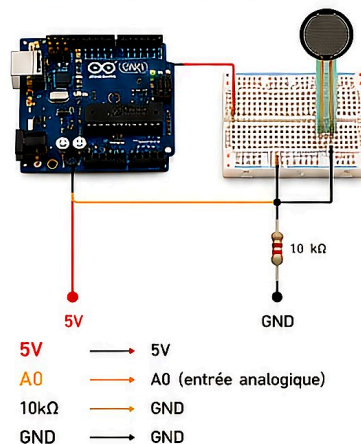
- Arduino Uno 1
- Capteur de force FSR 1
- Résistance 10 kΩ 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1



i La résistance de 10 kΩ forme un diviseur de tension avec le FSR pour lire la variation.

SCHEMA DE CÂBLAGE

Branchez le capteur de pression FSR comme suit :



i La broche A0 lira une tension entre 0 et 5 V selon la force appliquée.

CODE ARDUINO

```
const int fsrPin = A0; // Broche du FSR
const int ledPin = 13; // LED d'indication

void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}

void loop() {
  int valeur = analogRead(fsrPin); // 0 à 1023
  float tension = valeur * (5.0 / 1023.0); // 0 à 5 V
  int force = map(valeur, 0, 1023, 0, 100); // Force 0 à 100 %

  Serial.print("Valeur : ");
  Serial.print(valeur);
  Serial.print(" | Tension : ");
  Serial.print(tension, 2);
  Serial.print(" V | Force : ");
  Serial.print(force);
  Serial.println(" %");

  if (force > 50) {
    digitalWrite(ledPin, HIGH); // Force forte
  } else {
    digitalWrite(ledPin, LOW); // Force faible
  }
  delay(200);
}
```

Ouvrez le Moniteur Série à 9600 bauds pour voir les valeurs.

EXPLICATION DU PROGRAMME

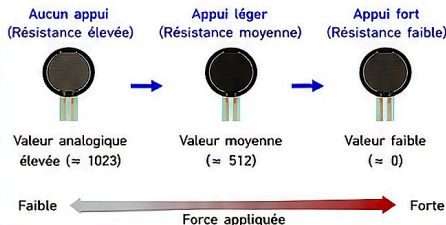
Le programme mesure la valeur analogique du FSR, la convertit en tension et en pourcentage de force.

Fonction	Rôle
<code>analogRead(fsrPin)</code>	Lit la valeur analogique (0 à 1023)
<code>map()</code>	Convertit la valeur en pourcentage
<code>Serial.print()</code>	Affiche les résultats
LED (pin 13)	S'allume si la force > 50 %

Vous pouvez ajuster le seuil (50 %) selon votre projet.

COMMENT FONCTIONNE LE FSR

Le FSR est composé d'un matériau semi-conducteur qui voit sa résistance diminuer lorsqu'on appuie dessus.

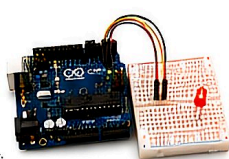


APPLICATIONS COURANTES

- Mesure de poids et balances électroniques
- Interfaces sensibles à la pression
- Détection de pas ou présence
- Contrôle de robot par pression
- Instruments de musique électroniques
- Projets interactifs et pédagogiques

EXEMPLE PRATIQUE : CONTRÔLE D'UNE LED SELON LA FORCE

- 1 Branchez le circuit comme indiqué.
- 2 Téléversez le code.
- 3 Appuyez doucement sur le FSR : la LED reste éteinte.
- 4 Appuyez fortement : la LED s'allume.
- 5 Observez les valeurs dans le Moniteur Série.



Sortie Moniteur Série (exemple)

Valeur : 120	Tension : 0.59 V	Force : 11 %
Valeur : 310	Tension : 1.51 V	Force : 38 %
Valeur : 620	Tension : 3.03 V	Force : 68 %
Valeur : 850	Tension : 4.15 V	Force : 83 %
Valeur : 980	Tension : 4.79 V	Force : 95 %

Conseil : Calibrez votre capteur en trouvant les valeurs min et max pour améliorer la précision.

À RETENIR

- ✓ Le FSR change de résistance selon la force appliquée.
- ✓ Utilisez une résistance de 10 kΩ pour former un diviseur de tension.
- ✓ Lisez la valeur analogique avec `analogRead()`.
- ✓ Vous pouvez convertir la valeur en tension et en pourcentage pour une meilleure interprétation.
- ✓ Idéal pour des projets interactifs et des mesures simples de pression ou de poids.



IDÉES D'AMÉLIORATION

- Affichez la force sur un écran LCD.
- Utilisez plusieurs FSR pour une surface de détection.
- Combinez avec un buzzer pour des alarmes de force.
- Ajoutez un enregistrement des données sur carte SD.



BON À SAVOIR

Le FSR n'est pas précis comme un capteur de pression industriel, mais il est économique, facile à utiliser et parfait pour l'apprentissage et les prototypes.



CONSEIL PRATIQUE

Ne pliez pas le FSR et n'appliquez pas une force supérieure à sa limite pour éviter de l'endommager. Testez dans différentes conditions pour calibrer vos mesures.

3

CHAPITRE 3

Arduino
et ESP32



Le DHT11 mesure la température et l'humidité relative de l'air. C'est un capteur numérique simple à utiliser pour les projets météo et domotiques.

3.22

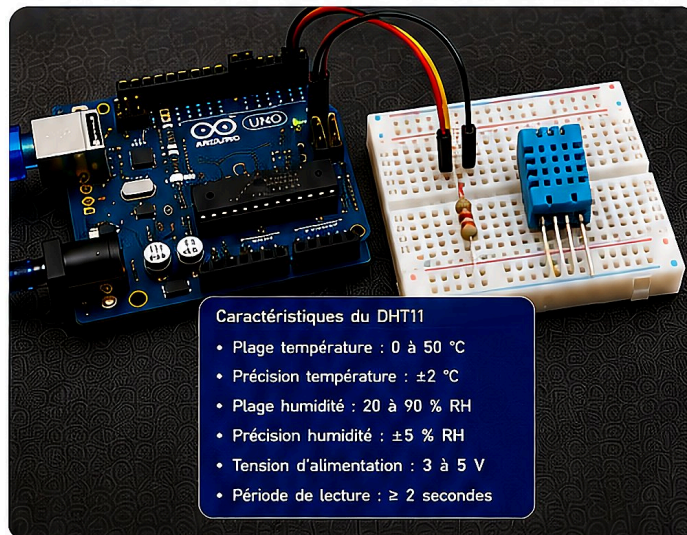
CAPTEUR DE TEMPÉRATURE ET D'HUMIDITÉ DHT11 AVEC ARDUINO

Surveiller l'environnement

Le DHT11 est un capteur numérique qui fournit la température en °C et l'humidité relative en %. Il utilise un seul fil de données pour communiquer avec Arduino, ce qui simplifie le câblage.



Le DHT11 ne nécessite qu'une brève communication toutes les 2 secondes. Évitez de le lire trop fréquemment pour garantir des mesures stables.

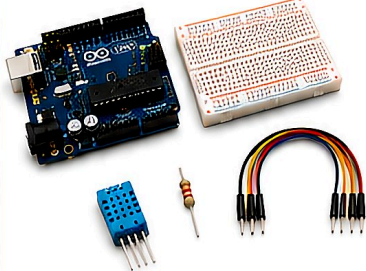


Caractéristiques du DHT11

- Plage température : 0 à 50 °C
- Précision température : ±2 °C
- Plage humidité : 20 à 90 % RH
- Précision humidité : ±5 % RH
- Tension d'alimentation : 3 à 5 V
- Période de lecture : ≥ 2 secondes

MATÉRIEL NÉCESSAIRE

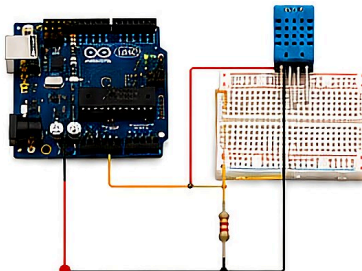
- Arduino Uno 1
- Capteur DHT11 1
- Résistance 10 kΩ 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1



La résistance de 10 kΩ est la résistance de tirage (pull-up) sur la broche DATA.

SCHEMA DE CÂBLAGE

Branchez le DHT11 comme suit :



- 5V → VCC (rouge)
 - D2 → DATA (jaune)
 - GND → GND (noir)
- (avec résistance 10 kΩ entre DATA et 5V)



Respectez la polarité : le DHT11 possède 4 broches (VCC, DATA, NC, GND).

CODE ARDUINO



```
#include <DHT.h>

#define DHTPIN 2 // Broche DATA
#define DHTTYPE DHT11 // Type de capteur

DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
  Serial.println("DHT11 - Test");
}

void loop() {
  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature(); // °C

  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Erreur de lecture du DHT11 !");
  } else {
    Serial.print("Humidite: ");
    Serial.print(humidity);
    Serial.print(" %\t ");
    Serial.print("Temperature: ");
    Serial.print(temperature);
    Serial.println(" °C");
  }
  delay(2000); // Attendre au moins 2 secondes
}
```



Ouvrez le Moniteur Série à 9600 bauds pour voir les mesures.

EXPLICATION DU PROGRAMME

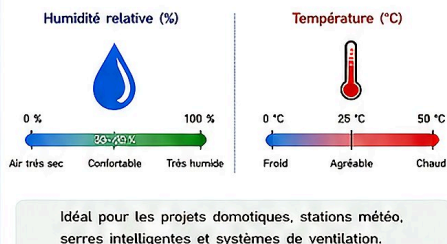
Le DHT11 envoie 40 bits de données à chaque lecture : 16 bits humidité + 16 bits température + 8 bits checksum.

Instruction	Rôle
dht.begin()	Initialise le capteur
readHumidity()	Lit l'humidité (%)
readTemperature()	Lit la température (°C)
isnan()	Vérifie si la lecture est valide
Serial.print()	Affiche les valeurs



Les lectures doivent être espacées d'au moins 2 secondes pour éviter des erreurs.

COMPRENDRE LES MESURES

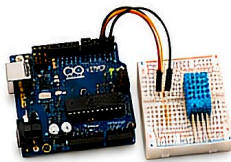


APPLICATIONS COURANTES

- Stations météo domestiques
- Contrôle d'humidité des serres
- Systèmes de climatisation et ventilation
- Surveillance de l'environnement intérieur
- Projets IoT et maisons intelligentes

EXEMPLE PRATIQUE : STATION MÉTÉO SIMPLE

- 1 Branchez le DHT11 selon le schéma.
- 2 Téléversez le code dans Arduino.
- 3 Ouvrez le Moniteur Série.
- 4 Observez les valeurs de température et d'humidité mises à jour toutes les 2 secondes.



Sortie Moniteur Série (exemple)

```
DHT11 - Test
Humidite: 45.0 % Temperature: 24.6 °C
Humidite: 44.8 % Temperature: 24.6 °C
Humidite: 45.1 % Temperature: 24.7 °C
Humidite: 45.0 % Temperature: 24.7 °C
Humidite: 44.9 % Temperature: 24.6 °C
...
```

Conseil : placez le capteur à l'abri du soleil direct et des sources de chaleur.



À RETENIR

- ✓ Le DHT11 mesure la température et l'humidité relative.
- ✓ Une résistance de 10 kΩ est nécessaire entre DATA et VCC.
- ✓ Lisez le capteur au maximum toutes les 2 secondes.
- ✓ Idéal pour surveiller l'environnement dans vos projets.



IDÉES D'AMÉLIORATION

- Enregistrez les données sur carte SD ou dans Firebase/ThingSpeak.
- Affichez les valeurs sur un écran LCD ou OLED.
- Ajoutez une alerte si la température ou l'humidité dépasse un seuil.



BON À SAVOIR

Le DHT11 est simple mais moins précis que le DHT22/AM2302. Pour plus de précision, utilisez le DHT22, mais le câblage et le code sont similaires.



CONSEIL PRATIQUE

Si vous obtenez des valeurs erronées (NaN), vérifiez les connexions, la résistance de pull-up et assurez-vous d'attendre 2 secondes entre chaque lecture.

3

CHAPITRE 3

Arduino
et ESP32



Le HC-SR04 utilise les ultrasons pour mesurer une distance sans contact. Idéal pour la détection d'obstacles et la robotique.

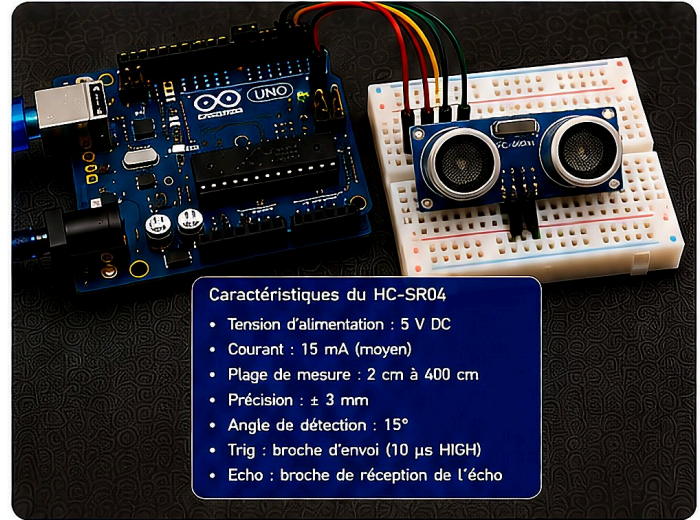
3.23

CAPTEUR À ULTRASONS HC-SR04 AVEC ARDUINO

Mesurer des distances avec précision

Le capteur HC-SR04 émet une onde ultrasonore et mesure le temps que prend l'écho pour revenir. En connaissant la vitesse du son, on peut calculer la distance entre le capteur et l'objet.

i La mesure dépend de la température de l'air et de la surface de l'objet. Évitez les surfaces molles ou inclinées pour plus de précision.

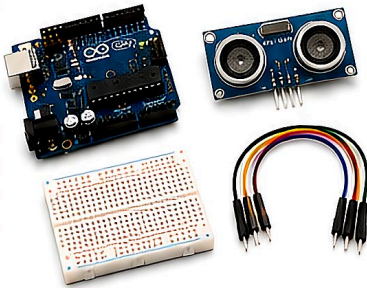


Caractéristiques du HC-SR04

- Tension d'alimentation : 5 V DC
- Courant : 15 mA (moyen)
- Plage de mesure : 2 cm à 400 cm
- Précision : ± 3 mm
- Angle de détection : 15°
- Trig : broche d'envoi (10 μ s HIGH)
- Echo : broche de réception de l'écho

MATÉRIEL NÉCESSAIRE

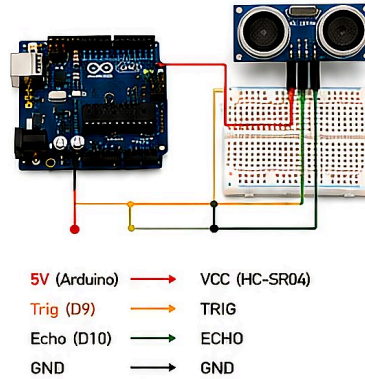
- Arduino Uno 1
- Capteur à ultrasons HC-SR04 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1



i Alimentez le module avec 5V uniquement. Ne pas dépasser cette tension.

SCHEMA DE CÂBLAGE

Branchez le HC-SR04 comme suit :



i Trig envoie l'impulsion de 10 μ s. Echo retourne la durée de l'écho.

CODE ARDUINO

```
const int trigPin = 9; // Broche TRIG
const int echoPin = 10; // Broche ECHO

void setup() {
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
  long duree;
  float distance;

  // Envoi d'une impulsion de 10  $\mu$ s sur TRIG
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Lecture de la durée de l'écho
  duree = pulseIn(echoPin, HIGH);

  // Calcul de la distance (en cm)
  distance = duree * 0.034 / 2;

  Serial.print("Distance : ");
  Serial.print(distance);
  Serial.println(" cm");
  delay(500);
}
```

Ouvrez le Moniteur Série à 9600 bauds pour voir les mesures.

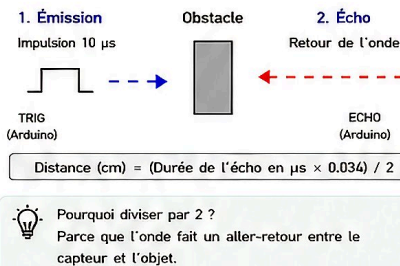
EXPLICATION DU PROGRAMME

Le HC-SR04 fonctionne en émettant une impulsion ultrasonore et en mesurant le temps de retour.

Instruction	Rôle
<code>digitalWrite(trigPin,HIGH)</code>	Envoie l'impulsion de 10 μ s
<code>pulseIn(echoPin, HIGH)</code>	Mesure la durée de l'écho
<code>distance = duree * 0.034 / 2</code>	Convertit le temps en distance
<code>delay(500)</code>	Attend 0.5 seconde entre chaque mesure

✓ La vitesse du son dans l'air ≈ 343 m/s
→ 0.034 cm/ μ s (à 20 °C)

COMMENT FONCTIONNE LE HC-SR04

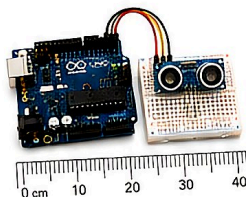


APPLICATIONS POSSIBLES

- Systèmes d'aide au stationnement
- Robots évitant les obstacles
- Mesure de niveau de remplissage
- Ouverture automatique de portes
- Alarmes de présence ou de sécurité

EXEMPLE PRATIQUE : DÉTECTION D'OBSTACLE

- 1 Téléversez le code sur l'Arduino.
- 2 Ouvrez le Moniteur Série.
- 3 Placez un objet devant le capteur.
- 4 Consultez la distance affichée en temps réel.



SORTIE MONITEUR SÉRIE (EXEMPLE)

Moniteur Série	
Distance :	35.12 cm
Distance :	34.87 cm
Distance :	34.95 cm
Distance :	20.31 cm
Distance :	15.67 cm
Distance :	10.02 cm
Distance :	7.05 cm
Distance :	5.03 cm

☑ Défilement auto Retour à la ligne 9600 baud

À RETENIR

- ✓ Le HC-SR04 mesure de 2 cm à 400 cm.
- ✓ Utilisez une impulsion de 10 μ s sur TRIG.
- ✓ La distance = (durée $\times$ 0.034) / 2.
- ✓ Évitez les surfaces molles ou inclinées.
- ✓ Idéal pour la robotique et la détection d'obstacles.

ATTENTION
Ne branchez pas VCC sur 3.3V. Utilisez uniquement 5V.

IDÉES D'AMÉLIORATION

- Ajoutez un écran LCD pour afficher la distance.
- Déclenchez une alarme ou une LED si l'objet est trop proche.
- Combinez plusieurs capteurs pour une meilleure couverture.
- Enregistrez les mesures sur une carte SD.

BON À SAVOIR

La précision dépend de la température de l'air. Pour plus de précision, vous pouvez compenser la température dans le calcul de la distance.

CONSEIL PRATIQUE

Testez toujours votre montage avec des objets à différentes distances pour comprendre le comportement du capteur.

3

CHAPITRE 3

Arduino
et ESP32



Un servomoteur permet un positionnement précis à un angle donné. Il est idéal pour la robotique, les bras mécaniques et les systèmes mobiles.

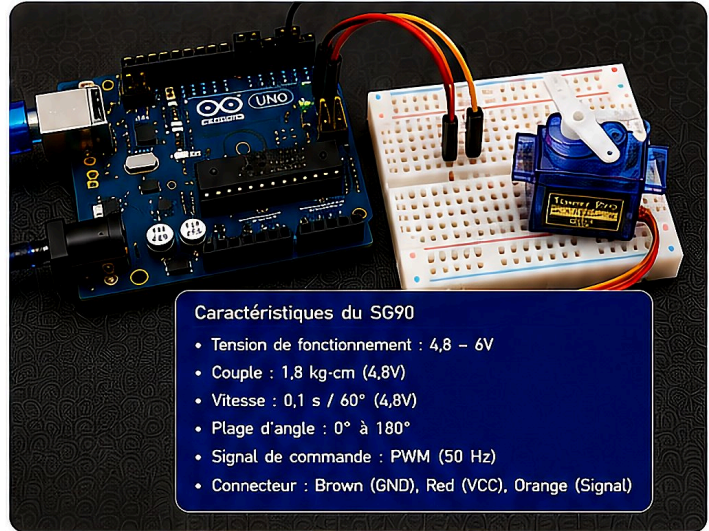
3.24

SERVOMOTEUR SG90 AVEC ARDUINO

Contrôler la position avec précision

Le servomoteur SG90 est un moteur à position contrôlée. On peut le faire tourner à un angle précis (de 0° à 180°) en envoyant un signal PWM (modulation de largeur d'impulsion).

i Utilisez une alimentation 5V externe si plusieurs servomoteurs sont utilisés pour éviter les chutes de tension sur l'Arduino.

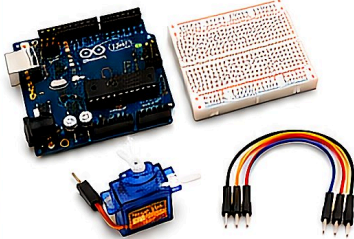


Caractéristiques du SG90

- Tension de fonctionnement : 4,8 – 6V
- Couple : 1,8 kg-cm (4,8V)
- Vitesse : 0,1 s / 60° (4,8V)
- Plage d'angle : 0° à 180°
- Signal de commande : PWM (50 Hz)
- Connecteur : Brown (GND), Red (VCC), Orange (Signal)

MATÉRIEL NÉCESSAIRE

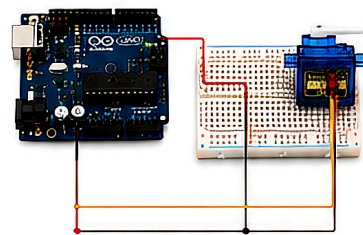
- Arduino Uno 1
- Servomoteur SG90 1
- Breadboard 1
- Fils de connexion Plusieurs
- Alimentation 5V externe (recommandée)
- Câble USB 1



i L'alimentation externe assure un fonctionnement stable du servomoteur.

SCHEMA DE CÂBLAGE

Branchez le servomoteur comme suit :



- 5V (externe) → Fil rouge du SG90
- D9 (Signal) → Fil orange du SG90
- GND → Fil marron du SG90
- GND commun entre l'Arduino et l'alimentation.

i Le fil de signal (orange) est connecté à la broche D9. Le GND doit être commun.

CODE ARDUINO

</>

```
#include <Servo.h> // Inclure la bibliothèque Servo
Servo monServo; // Créer un objet servo

void setup() {
  monServo.attach(9); // Attacher le servo à la broche D9
}

void loop() {
  // Aller de 0° à 180°
  for (int angle = 0; angle <= 180; angle++) {
    monServo.write(angle);
    delay(15); // Attendre 15 ms
  }
  delay(1000); // Pause 1 seconde

  // Revenir de 180° à 0°
  for (int angle = 180; angle >= 0; angle--) {
    monServo.write(angle);
    delay(15);
  }
  delay(1000); // Pause 1 seconde
}
```

i Ouvrez le Moniteur Série à 9600 bauds pour voir les messages (optionnel).

EXPLICATION DU PROGRAMME

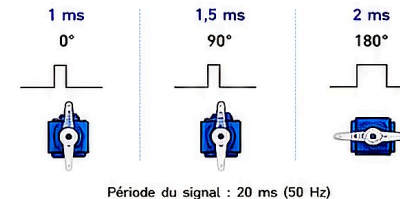
Le programme fait tourner le servomoteur doucement de 0° à 180° puis revient à 0° en continu.

Fonction	Rôle
Servo monServo;	Crée un objet servo
monServo.attach(9);	Indique la broche de contrôle (D9)
monServo.write(angle);	Positionne le servo à l'angle donné (0-180°)
delay(15);	Laisse le temps au servo d'atteindre la position
delay(1000);	Pause d'une seconde entre les cycles

✓ Le signal PWM est envoyé toutes les 20 ms (50 Hz), avec des impulsions de 1 à 2 ms.

COMMENT FONCTIONNE UN SERVOMOTEUR

Le servomoteur reçoit un signal PWM. La durée de l'impulsion détermine l'angle de rotation.



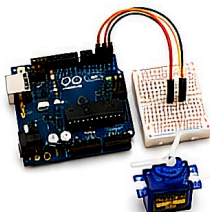
✓ Une impulsion plus longue = angle plus grand.

APPLICATIONS COURANTES

- Bras robotiques et manipulateurs
- Caméras motorisées (pan/tilt)
- Modèles réduits (avions, voitures, bateaux)
- Ouverture de portes ou volets automatiques
- Projets éducatifs et démonstrations

EXEMPLE PRATIQUE : BALAYAGE 0° - 180°

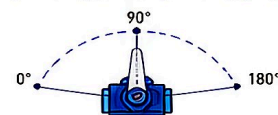
- 1 Branchez le servomoteur SG90 selon le schéma.
- 2 Téléversez le code sur l'Arduino.
- 3 Le servo tourne de 0° à 180° progressivement.
- 4 Il attend 1 seconde.
- 5 Il revient ensuite de 180° à 0°.



Sortie attendue (intersectionnel)

Le servomoteur se déplace continuellement :

0° → 180° → 0° → 180° ...



Conseil : Si le servo vibre ou ne se positionne pas correctement, vérifiez l'alimentation et le câblage (masse commune).

À RETENIR

- ✓ Le servomoteur est contrôlé par un signal PWM.
- ✓ Les angles valides sont de 0° à 180°.
- ✓ Utilisez une alimentation 5V stable (surtout pour plusieurs servos).
- ✓ La broche de signal peut être modifiée facilement dans le code.



ATTENTION

Ne jamais alimenter le servo directement depuis la broche 5V de l'Arduino. Cela peut endommager la carte.



IDÉES D'AMÉLIORATION

- Contrôlez le servo avec un potentiomètre.
- Utilisez plusieurs servos pour un bras robotique.
- Ajoutez des boutons pour mémoriser des positions.
- Commandez-le depuis un écran LCD ou via Bluetooth.



BON À SAVOIR

Les servomoteurs utilisent des engrenages et un capteur de position interne pour maintenir l'angle souhaité. Ils corrigent automatiquement leur position grâce au signal reçu.



CONSEIL PRATIQUE

Testez toujours le servo sans charge avant de l'intégrer à un mécanisme plus lourd pour vérifier la plage de mouvement et le couple.

3

CHAPITRE 3

Arduino
et ESP32



Un afficheur LCD permet de présenter des messages, des mesures et des états en temps réel. C'est un élément essentiel pour rendre vos projets interactifs et professionnels.

3.25

AFFICHEUR LCD 16x2 AVEC ARDUINO

Afficher des informations claires

L'écran LCD 16x2 permet d'afficher du texte sur 2 lignes de 16 caractères. Il est idéal pour les projets de mesure, de contrôle et d'interface utilisateur.

i Le LCD communique avec l'Arduino en mode parallèle (4 bits) pour économiser les broches.

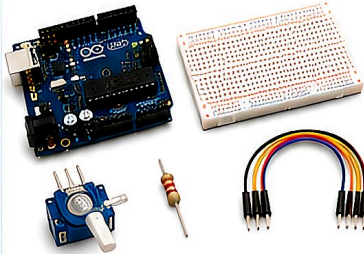


Caractéristiques du LCD 16x2

- 16 caractères par ligne, 2 lignes
- Tension de fonctionnement : 5V
- Courant : ~1 mA (sans rétroéclairage)
- Contrôle en mode 4 bits ou 8 bits
- Rétroéclairage LED intégré
- Contraste réglable via potentiomètre

MATÉRIEL NÉCESSAIRE

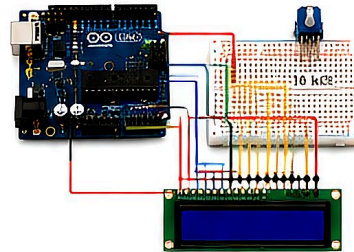
- Arduino Uno 1
- Afficheur LCD 16x2 (HD44780) 1
- Potentiomètre 10 kΩ 1
- Breadboard 1
- Fils de connexion Plusieurs
- Résistance 220 Ω 1
- Câble USB 1



i La résistance (220 Ω) limite le courant du rétroéclairage pour protéger le LCD.

SCHÉMA DE CÂBLAGE

Branchez le LCD 16x2 comme suit (mode 4 bits) :



LCD	Arduino Uno
VSS (1)	GND
VDD (2)	5V
VO (3)	Curseur du potentiomètre
RS (4)	D12
RW (5)	GND
E (6)	D11
D4 (11)	D5
D5 (12)	D4
D6 (13)	D3
D7 (14)	D2
A (15)	5V (via 220 Ω)
K (16)	GND

i Réglez le potentiomètre pour obtenir le meilleur contraste de l'affichage.

CODE ARDUINO

```
#include <LiquidCrystal.h>

// Définition des broches
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  lcd.begin(16, 2); // Initialiser LCD 16x2
  lcd.clear(); // Effacer l'écran
  lcd.setCursor(0, 0); // Colonne 0, ligne 0
  lcd.print("DEX ATOMES");
  lcd.setCursor(0, 1); // Colonne 0, ligne 1
  lcd.print("LCD 16x2 Test");
}

void loop() {
  // Exemple : afficher un compteur
  for (int i = 0; i <= 99; i++) {
    lcd.setCursor(0, 0);
    lcd.print("Compteur : ");
    if (i < 10) lcd.print("0");
    lcd.print(i);
    lcd.print(" ");
    lcd.print(" "); // Efface le reste
    delay(1000); // Attendre 1 seconde
  }
}
```

i Ouvrez le Moniteur Série à 9600 bauds (facultatif, ce code n'affiche rien dans le moniteur).

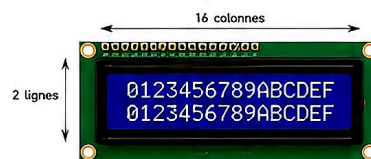
EXPLICATION DU PROGRAMME

Le code initialise le LCD, affiche un message fixe puis un compteur qui s'incrémente de 0 à 99.

Instruction	Rôle
lcd.begin(16, 2)	Initialise l'écran 16x2
lcd.clear()	Efface l'affichage
lcd.setCursor(col, ligne)	Place le curseur
lcd.print("texte")	Affiche du texte
delay(1000)	Attend 1 seconde

✓ Tous les caractères affichés doivent être en majuscules ou minuscules selon vos besoins.

COMPRENDRE L'AFFICHEUR 16x2



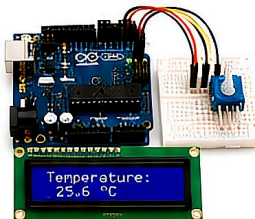
- Chaque ligne contient 16 caractères.
- Les positions commencent à (0,0) en haut à gauche.
- Origine : colonne 0 à 15, ligne 0 ou 1.

APPLICATIONS COURANTES

- Affichage de mesures (température, distance)
- Interfaces utilisateur simples
- Indicateurs d'état de systèmes
- Menus dans les projets interactifs
- Projets éducatifs et démonstrations

EXEMPLE PRATIQUE : THERMOMÈTRE NUMÉRIQUE

- 1 Branchez le capteur de température (LM35 ou DHT11).
- 2 Téléversez le code correspondant.
- 3 L'écran affiche la température en temps réel.
- 4 Réglez le contraste avec le potentiomètre.
- 5 Alimentez en 5V et observez !



SORTIE EXEMPLE SUR LCD

DEX ATOMES LCD 16x2 Test
Compteur : 07
Compteur : 42
Compteur : 99

À RETENIR

- ✓ Le LCD 16x2 rend vos projets plus professionnels et informatifs.
- ✓ Utilisez le mode 4 bits pour économiser les broches.
- ✓ Réglez toujours le contraste pour une lecture optimale.
- ✓ Combinez-le avec des capteurs pour créer des systèmes intelligents.



IDÉES D'AMÉLIORATION

- Ajoutez un rétroéclairage RGB pour plus de style.
- Créez un menu navigable avec des boutons.
- Enregistrez des données sur carte SD et affichez-les.



BON À SAVOIR

Le contrôleur le plus courant est le HD44780, compatible avec la majorité des écrans LCD 16x2 du marché.



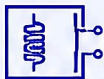
CONSEIL PRATIQUE

Évitez d'écrire trop souvent au LCD pour éviter un scintillement. Utilisez des délais raisonnables dans vos boucles.

3

CHAPITRE 3

Arduino
et ESP32



Le relais agit comme un interrupteur contrôlé électriquement. Il permet à Arduino de commander des appareils fonctionnant en haute tension ou à fort courant en toute sécurité.

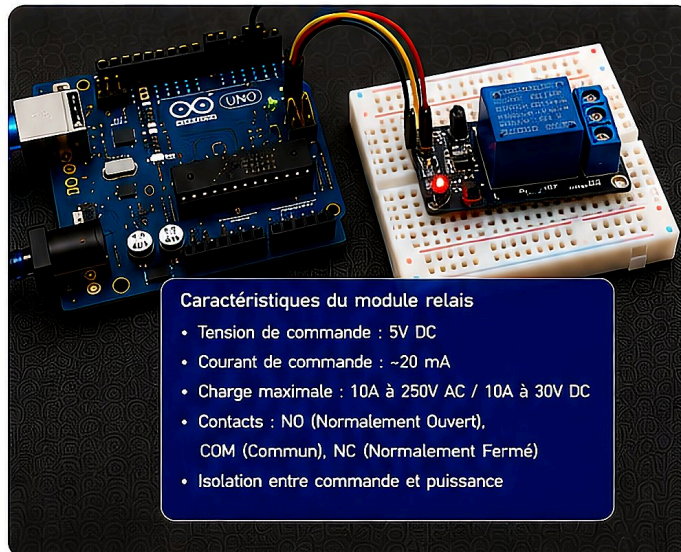
3.26

MODULE RELAIS 1 CANAL AVEC ARDUINO

Commander des appareils électriques facilement

Le module relais 1 canal permet à Arduino de contrôler un appareil alimenté en 110V/220V AC ou en 12V/24V DC. Le relais isole la partie puissance de la partie commande pour protéger votre carte et vos circuits.

i Ne branchez jamais la tension secteur directement sur l'Arduino. Utilisez toujours le module relais.



Caractéristiques du module relais

- Tension de commande : 5V DC
- Courant de commande : ~20 mA
- Charge maximale : 10A à 250V AC / 10A à 30V DC
- Contacts : NO (Normalement Ouvert), COM (Commun), NC (Normalement Fermé)
- Isolation entre commande et puissance

MATÉRIEL NÉCESSAIRE

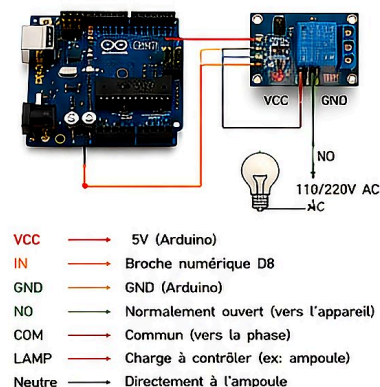
- Arduino Uno 1
- Module relais 1 canal 5V 1
- Breadboard 1
- Fils de connexion Plusieurs
- Ampoule 110V (ou autre charge) 1
- Câble secteur + prise 1



i Utilisez le module relais pour contrôler des appareils en toute sécurité.

SCHEMA DE CÂBLAGE

Branchez le module relais comme suit :



i Le neutre de l'ampoule est connecté directement au secteur. Seule la phase passe par le relais.

CODE ARDUINO

```
const int relaisPin = 8; // Broche de commande du relais

void setup() {
  pinMode(relaisPin, OUTPUT); // Configure la broche en sortie
  digitalWrite(relaisPin, LOW); // Relais désactivé au démarrage
  Serial.begin(9600);
  Serial.println("Systeme Relais - Pret");
}

void loop() {
  Serial.println("Relais ON (ampoule allumee)");
  digitalWrite(relaisPin, HIGH); // Active le relais
  delay(5000); // Attendre 5 secondes

  Serial.println("Relais OFF (ampoule eteinte)");
  digitalWrite(relaisPin, LOW); // Désactive le relais
  delay(5000); // Attendre 5 secondes
}
```

Ouvrez le Moniteur Série à 9600 bauds pour suivre l'état du relais.

EXPLICATION DU PROGRAMME

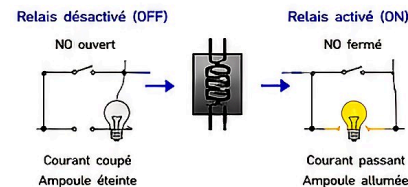
Le programme active le relais pendant 5 secondes puis le désactive pendant 5 secondes en boucle.

Instruction	Rôle
<code>pinMode(relaisPin, OUTPUT)</code>	Configure D8 en sortie
<code>digitalWrite(relaisPin, HIGH)</code>	Active le relais (ON)
<code>digitalWrite(relaisPin, LOW)</code>	Désactive le relais (OFF)
<code>delay(5000)</code>	Attente 5 secondes

✓ Quand le relais est activé, le contact NO se ferme et l'ampoule s'allume.

COMMENT FONCTIONNE LE RELAIS

Le relais agit comme un interrupteur commandé.



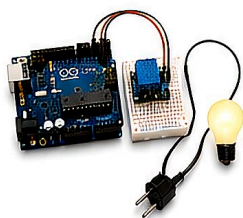
i Le circuit de commande (5V) est isolé du circuit de puissance (110/220V).

APPLICATIONS COURANTES

- Éclairage automatique
- Pompes et systèmes d'irrigation
- Ventilateurs et climatiseurs
- Ouverture de barrières ou portails
- Domotique et maisons intelligentes
- Projets industriels et pédagogiques

EXEMPLE PRATIQUE : ALLUMAGE D'UNE AMPOULE

- 1 Branchez le circuit selon le schéma.
- 2 Téléversez le code sur l'Arduino.
- 3 L'ampoule s'allume pendant 5 secondes.
- 4 Elle s'éteint pendant 5 secondes.
- 5 Le cycle se répète indéfiniment.



SORTIE MONITEUR SÉRIE (EXEMPLE)

```
Systeme Relais - Pret
Relais ON (ampoule allumee)
Relais OFF (ampoule eteinte)
Relais ON (ampoule allumee)
Relais OFF (ampoule eteinte)
Relais ON (ampoule allumee)
...
```

Conseil : Le délai peut être modifié pour des applications différentes.

À RETENIR

- ✓ Le relais permet à Arduino de contrôler des charges fortes en toute sécurité.
- ✓ Utilisez toujours le module relais pour séparer la commande de la puissance.
- ✓ Vérifiez les connexions NO, NC et COM avant la mise sous tension.
- ✓ Ne touchez jamais les fils 110/220V pendant le fonctionnement.



ATTENTION
Travaillez toujours avec prudence sur le secteur. Coupez l'alimentation avant toute manipulation.



IDÉES D'AMÉLIORATION

- Ajoutez un bouton pour activer/désactiver manuellement.
- Contrôlez le relais depuis une application mobile (Wi-Fi).
- Utilisez un écran LCD pour afficher l'état du relais.
- Créez un minuteur ou un programmeur d'événements.



BON À SAVOIR

Le relais est un composant indispensable pour vos projets de domotique, robotique et automatisation industrielle.



CONSEIL PRATIQUE

Testez toujours votre montage avec une petite charge avant de connecter des appareils de forte puissance.

3

CHAPITRE 3

Arduino
et ESP32



Un moteur pas à pas se déplace par petits incréments appelés pas. Il permet un positionnement précis sans capteur de rétroaction.

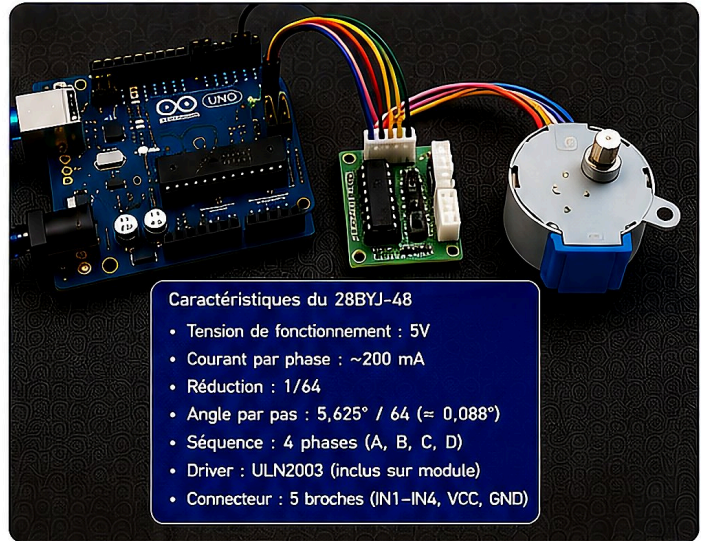
3.27

MOTEUR PAS À PAS 28BYJ-48 AVEC DRIVER ULN2003 ET ARDUINO

Contrôler la position pas à pas

Le moteur 28BYJ-48 est léger, économique et facile à utiliser avec le driver ULN2003. Il est idéal pour les projets de positionnement précis comme les bras robotiques, caméras ou maquettes automatiques.

i Le driver ULN2003 contient 7 transistors Darlington pour piloter les 4 bobines du moteur. Alimentez l'Arduino en 5V et le moteur en 5V.

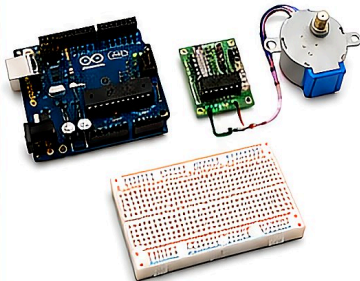


Caractéristiques du 28BYJ-48

- Tension de fonctionnement : 5V
- Courant par phase : ~200 mA
- Réduction : 1/64
- Angle par pas : $5,625^\circ / 64 \approx 0,088^\circ$
- Séquence : 4 phases (A, B, C, D)
- Driver : ULN2003 (inclus sur module)
- Connecteur : 5 broches (IN1–IN4, VCC, GND)

MATÉRIEL NÉCESSAIRE

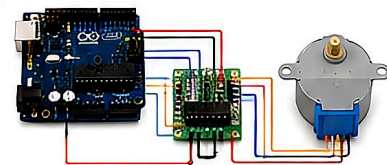
- Arduino Uno 1
- Moteur pas à pas 28BYJ-48 1
- Driver ULN2003 1
- Breadboard 1
- Fils de connexion Plusieurs
- Alimentation 5V externe (recommandée) 1
- Câble USB 1



i Une alimentation 5V externe est recommandée si vous alimentez plusieurs moteurs.

SCHEMA DE CÂBLAGE

Branchez le moteur comme suit :



- IN1 (ULN2003) → D8 (Arduino)
- IN2 (ULN2003) → D9 (Arduino)
- IN3 (ULN2003) → D10 (Arduino)
- IN4 (ULN2003) → D11 (Arduino)
- VCC (ULN2003) → 5V (externe ou Arduino)
- GND (ULN2003) → GND (Arduino et 5V externe)
- Moteur (5 fils) → Connecteur du 28BYJ-48

i Ne pas inverser VCC et GND. Reliez les masses (GND) de l'Arduino et de l'alimentation externe si utilisée.

CODE ARDUINO

```
#include <Stepper.h>

const int stepsPerRevolution = 2048; // 32 x 64
Stepper myStepper(stepsPerRevolution, 8, 10, 9, 11); // IN1, IN3, IN2, IN4

void setup() {
  myStepper.setSpeed(10); // Vitesse en RPM
  Serial.begin(9600);
  Serial.println("Moteur pas à pas 28BYJ-48 Test");
}

void loop() {
  Serial.println("Rotation horaire 1 tour");
  myStepper.step(stepsPerRevolution); // 1 tour
  delay(1000);

  Serial.println("Rotation antihoraire 1 tour");
  myStepper.step(-stepsPerRevolution); // -1 tour
  delay(1000);

  Serial.println("Avance de 512 pas");
  myStepper.step(512);
  delay(1000);

  Serial.println("Recul de 512 pas");
  myStepper.step(-512);
  delay(1000);
}
```

i Ouvrez le Moniteur Série à 9600 bauds pour suivre les mouvements.

EXPLICATION DU PROGRAMME

La bibliothèque Stepper.h simplifie le contrôle des moteurs pas à pas 4 fils via le driver ULN2003.

Commande	Rôle
<code>myStepper.setSpeed(rpm);</code>	Définit la vitesse (tours/minute)
<code>myStepper.step(n);</code>	Fait tourner le moteur de n pas (n positif ou négatif)
<code>stepsPerRevolution</code>	Nombre de pas pour 1 tour (2048)

✓ Le moteur effectue des mouvements complets, puis des mouvements partiels pour démonstration.

SÉQUENCE DE PILOTAGE (4 PHASES)

Le driver active les bobines dans cet ordre pour faire tourner le moteur dans un sens.

Pas	IN1 (A)	IN2 (B)	IN3 (C)	IN4 (D)
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

(Se répète)

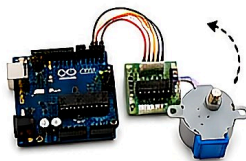
i Inverser l'ordre des phases inverse le sens de rotation.

APPLICATIONS COURANTES

- Bras robotiques et actionneurs
- Caméras motorisées (pan/tilt)
- Imprimantes 3D et CNC
- Distributeurs automatiques
- Maquettes et projets éducatifs

EXEMPLE PRATIQUE : CONTRÔLE DE POSITION

- 1 Téléversez le code sur l'Arduino.
 - 2 Le moteur tourne 1 tour dans un sens.
 - 3 Puis 1 tour dans l'autre sens.
 - 4 Ensuite il avance de 512 pas.
 - 5 Puis recule de 512 pas.
- Répétez la séquence.



SORTIE MONITEUR SÉRIE (EXEMPLE)

Moteur pas à pas 28BYJ-48 Test
Rotation horaire 1 tour
Rotation antihoraire 1 tour
Avance de 512 pas
Recul de 512 pas
Rotation horaire 1 tour
Rotation antihoraire 1 tour
...

La sortie affiche l'état actuel du programme.

À RETENIR

- ✓ Utilisez une alimentation 5V stable si possible.
- ✓ Ne connectez jamais le 5V moteur sans vérifier le courant requis.
- ✓ Le 28BYJ-48 est précis, silencieux et parfait pour les projets éducatifs.
- ✓ Le driver ULN2003 simplifie la connexion et protège l'Arduino.

ATTENTION
Ne tirez pas sur les fils du connecteur du moteur. Débranchez l'alimentation avant de modifier le câblage.

IDÉES D'AMÉLIORATION

- Ajoutez des fins de course pour limiter la course.
- Utilisez un encodeur pour une précision accrue.
- Intégrez avec un écran LCD pour afficher la position.
- Contrôlez via Bluetooth ou Wi-Fi.

BON À SAVOIR

Le 28BYJ-48 consomme peu d'énergie et peut être contrôlé directement par l'Arduino via le driver. Idéal pour l'apprentissage de la robotique et de l'automatisation.

CONSEIL PRATIQUE

Testez vos mouvements à faible vitesse avant d'augmenter. Utilisez des pauses (delay) pour observer le comportement du moteur.

3

CHAPITRE 3

Arduino
et ESP32



Le joystick permet de détecter la position dans 2 directions (X, Y) et l'appui sur le bouton. Il est idéal pour contrôler des robots, des menus ou des jeux.

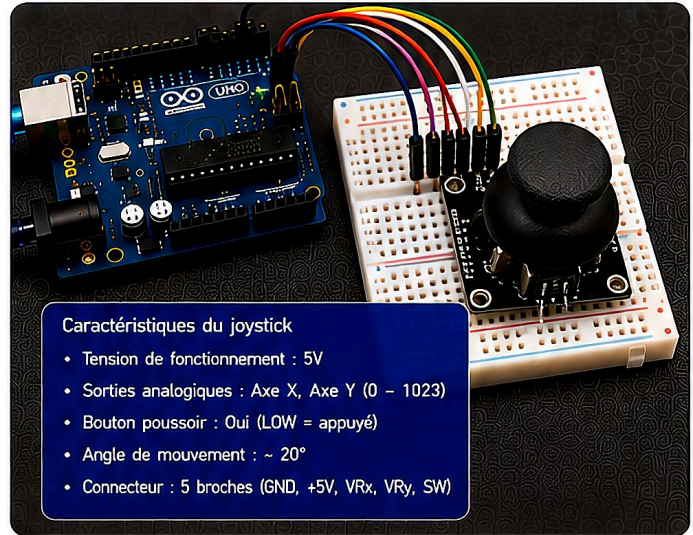
3.28

JOYSTICK ANALOGIQUE AVEC ARDUINO

Lire les axes X et Y et le bouton

Le joystick analogique possède deux potentiomètres utilisés pour les axes X et Y, ainsi qu'un bouton poussoir intégré. Avec l'Arduino, on peut lire ces valeurs et réagir en conséquence.

i Les axes X et Y renvoient des valeurs analogiques de 0 à 1023. Le bouton donne un état HIGH (non appuyé) ou LOW (appuyé).

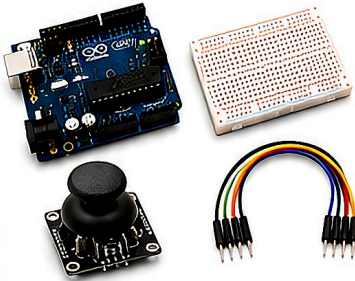


Caractéristiques du joystick

- Tension de fonctionnement : 5V
- Sorties analogiques : Axe X, Axe Y (0 - 1023)
- Bouton poussoir : Oui (LOW = appuyé)
- Angle de mouvement : ~ 20°
- Connecteur : 5 broches (GND, +5V, VRx, VRy, SW)

MATÉRIEL NÉCESSAIRE

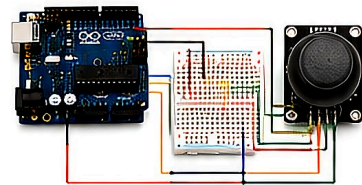
- Arduino Uno 1
- Joystick analogique 1
- Breadboard 1
- Fils de connexion Plusieurs
- Câble USB 1



i Alimentez l'Arduino en 5V pour lire correctement les valeurs analogiques.

SCHEMA DE CÂBLAGE

Branchez le joystick comme suit :



Broche Joystick	Fonction	Arduino Uno
GND	Masse	GND
+5V	Alimentation	5V
VRx	Axe X (analogique)	A0
VRy	Axe Y (analogique)	A1
SW	Bouton (poussoir)	D2

i Le bouton est câblé sur D2 avec la résistance interne de pull-up activée dans le code.

CODE ARDUINO

```
const int xPin = A0; // Axe X
const int yPin = A1; // Axe Y
const int buttonPin = 2; // Bouton du joystick

void setup() {
  Serial.begin(9600);
  pinMode(buttonPin, INPUT_PULLUP); // Bouton avec pull-up interne
  Serial.println("X, Y, Bouton");
}

void loop() {
  int xValue = analogRead(xPin); // Lecture Axe X
  int yValue = analogRead(yPin); // Lecture Axe Y
  int buttonState = digitalRead(buttonPin); // Lecture bouton

  Serial.print(xValue);
  Serial.print(",");
  Serial.print(yValue);
  Serial.print(",");
  Serial.println(buttonState == LOW ? "Appuyé" : "Relâché");

  delay(100);
}
```

Ouvrez le Moniteur Série à 9600 bauds pour voir les valeurs en temps réel.

EXPLICATION DU PROGRAMME

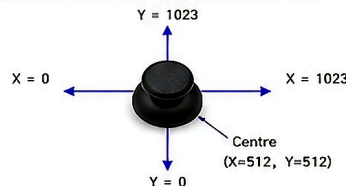
Le programme lit en continu les positions X et Y du joystick ainsi que l'état du bouton et les envoie au Moniteur Série.

Commande	Rôle
<code>analogRead(A0)</code>	Lit la valeur de l'axe X (0 à 1023)
<code>analogRead(A1)</code>	Lit la valeur de l'axe Y (0 à 1023)
<code>digitalRead(D2)</code>	Lit l'état du bouton (LOW si appuyé)
<code>INPUT_PULLUP</code>	Active la résistance de tirage interne

✓ Les valeurs proches de 512 (milieu) indiquent que le joystick est au centre.

COMPRENDRE LES AXES

Le joystick renvoie des valeurs entre 0 et 1023.



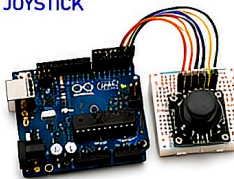
Poussez le joystick à fond dans une direction pour obtenir 0 ou 1023 sur l'axe correspondant.

APPLICATIONS POSSIBLES

- Contrôle de robots et de bras mécaniques
- Navigation dans les menus d'un écran LCD
- Jeux interactifs et consoles DIY
- Contrôle de caméras (pan/tilt)
- Projets de drones et véhicules télécommandés

EXEMPLE PRATIQUE : LIRE LE JOYSTICK

- 1 Téléversez le code sur l'Arduino.
- 2 Ouvrez le Moniteur Série.
- 3 Déplacez le joystick dans toutes les directions.
- 4 Appuyez sur le bouton.
- 5 Observez les valeurs affichées.



SORTIE MONITEUR SÉRIE (EXEMPLE)

```
Moniteur Série
X, Y, Bouton
512, 512, Relâché
820, 212, Relâché
512, 210, Appuyé
150, 512, Relâché
512, 900, Relâché
```

✓ Défilement auto Retour à la ligne 9600 baud

À RETENIR

- ✓ Les axes X et Y renvoient des valeurs de 0 à 1023. 512 correspond au centre.
- ✓ Le bouton est à LOW lorsqu'il est appuyé.
- ✓ Utilisez `INPUT_PULLUP` pour simplifier le câblage.
- ✓ Le joystick est un excellent outil d'interface utilisateur.



ATTENTION
Ne forcez pas le joystick au-delà de ses butées mécaniques pour éviter de l'endommager.



IDÉES D'AMÉLIORATION

- Ajoutez un écran LCD pour afficher les valeurs X et Y.
- Utilisez la valeur du joystick pour contrôler la vitesse d'un moteur.
- Combinez plusieurs joysticks pour un contrôle avancé.



BON À SAVOIR

Les valeurs analogiques peuvent varier légèrement selon la qualité du joystick. Calibrez si nécessaire pour plus de précision.



CONSEIL PRATIQUE

Testez toujours votre montage et votre code dans le Moniteur Série avant de l'intégrer dans votre projet final.

3

CHAPITRE 3

Arduino
et ESP32



Le potentiomètre est une résistance variable. Il permet de régler une valeur manuellement pour contrôler un paramètre (LED, vitesse, son, etc.).

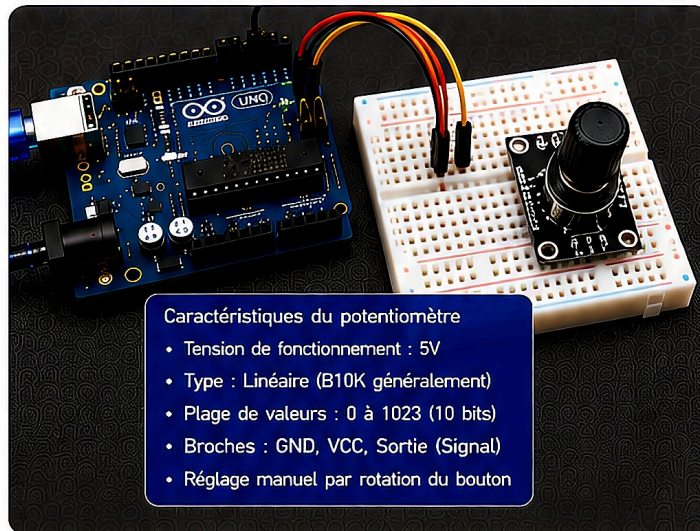
3.29

POTENTIOMÈTRE AVEC ARDUINO

Lire une valeur analogique

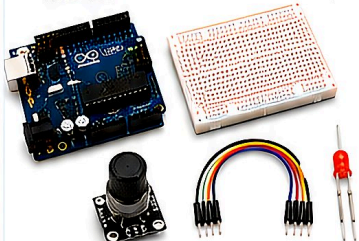
Le potentiomètre est un composant simple et très utile pour régler manuellement des valeurs. L'Arduino lit la position du curseur et peut utiliser cette valeur pour contrôler des sorties.

i La valeur lue varie de 0 (bout à gauche) à 1023 (bout à droite).



MATÉRIEL NÉCESSAIRE

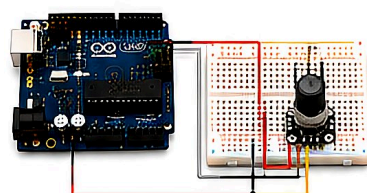
- Arduino Uno 1
- Potentiomètre 10 kΩ (B10K) 1
- Breadboard 1
- Fils de connexion Plusieurs
- LED (optionnel) 1
- Résistance 220 Ω (optionnel) 1
- Câble USB 1



i Le potentiomètre possède 3 broches : GND – Sortie (curseur) – VCC (5V).

SCHEMA DE CÂBLAGE

Branchez le potentiomètre comme suit :



- 5V (Arduino) → VCC du potentiomètre
- A0 (Arduino) → Sortie (curseur)
- GND (Arduino) → GND du potentiomètre

i La broche centrale (curseur) est reliée à l'entrée analogique A0.

CODE ARDUINO



```
void setup() {
  Serial.begin(9600); // Démarrer la communication série
}

void loop() {
  int valeur = analogRead(A0); // Lire la valeur du potentiomètre
  Serial.print("Valeur : ");
  Serial.println(valeur); // Afficher la valeur (0 à 1023)
  delay(200); // Attendre 200 ms
}
```

Ouvrez le Moniteur Série à 9600 bauds pour voir les valeurs en temps réel.

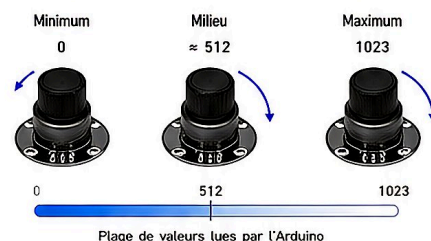
EXPLICATION DU PROGRAMME

Le code lit la valeur analogique sur A0 et l'affiche dans le Moniteur Série toutes les 200 ms.

Ligne de code	Rôle
<code>analogRead(A0)</code>	Lit la valeur du potentiomètre (0 à 1023)
<code>Serial.print()</code>	Affiche un texte fixe
<code>Serial.println(valeur)</code>	Affiche la valeur lue
<code>delay(200)</code>	Attendre 200 millisecondes

✓ Plus le bouton est tourné vers la droite, plus la valeur est grande.

COMMENT FONCTIONNE LE POTENTIOMÈTRE



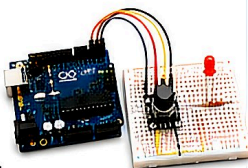
Tournez le bouton lentement et observez la valeur changer dans le Moniteur Série.

APPLICATIONS COURANTES

- Réglage de la luminosité d'une LED
- Contrôle de la vitesse d'un moteur
- Volume sonore dans un projet audio
- Navigation dans un menu LCD
- Commandes manuelles et interfaces utilisateur

EXEMPLE PRATIQUE : CONTRÔLE DE LUMINOSITÉ LED

- 1 Branchez le potentiomètre comme indiqué ci-dessus.
- 2 Connectez une LED via la résistance de 220 Ω sur la broche 9.
- 3 Téléversez le code ci-contre.
- 4 Tournez le bouton du potentiomètre.
- 5 La LED devient plus ou moins brillante !



CODE : LUMINOSITÉ LED

```
int ledPin = 9;

void setup() {
  pinMode(ledPin, OUTPUT);
}

void loop() {
  int valeur = analogRead(A0); // 0 à 1023
  int luminosite = map(valeur, 0, 1023, 0, 255);
  analogWrite(ledPin, luminosite); // PWM sur D9
  delay(20);
}
```

✓ La fonction `map()` convertit la valeur 0-1023 en 0-255 pour la luminosité (PWM).

À RETENIR

- ✓ Le potentiomètre est une résistance variable.
- ✓ Sa valeur varie de 0 à 1023 avec l'Arduino.
- ✓ La broche du milieu est la sortie (signal).
- ✓ Utilisez-le pour ajuster manuellement des paramètres dans vos projets.

ATTENTION
Ne forcez pas le potentiomètre au-delà de ses limites mécaniques. Toujours connecter VCC, GND et Signal correctement.



IDÉES D'AMÉLIORATION

- Ajoutez un écran LCD pour afficher la valeur.
- Contrôlez un servo moteur avec le potentiomètre.
- Combinez plusieurs potentiomètres pour plus de contrôles.



BON À SAVOIR

Le potentiomètre linéaire est le plus courant. Il existe aussi des modèles logarithmiques (utilisés pour le son).



CONSEIL PRATIQUE

Filtrez la valeur avec un petit délai (20-50 ms) pour éviter les variations trop rapides ou bruitées.

3

CHAPITRE 3 Arduino et ESP32



Ce robot détecte les obstacles, prend une décision et se déplace automatiquement pour les éviter.

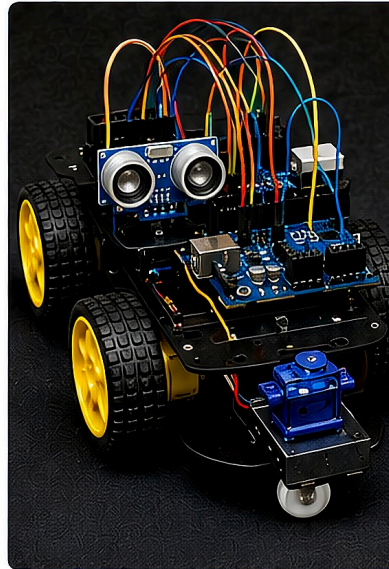
3.30

PROJET FINAL ROBOT ÉVITANT LES OBSTACLES AVEC ARDUINO

Construisons un robot autonome capable de détecter les obstacles et d'éviter les collisions en temps réel.

Ce projet combine plusieurs composants que nous avons étudiés : capteur ultrason, servomoteur, moteurs à courant continu, driver L298N et Arduino. Il constitue une excellente application pratique de la robotique.

- i** Le capteur HC-SR04 mesure la distance devant le robot. Si un obstacle est détecté à moins de 20 cm, le robot s'arrête, regarde à gauche et à droite, puis choisit la direction la plus libre.



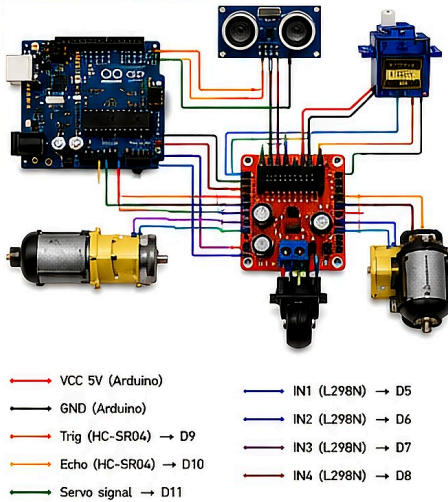
COMPOSANTS UTILISÉS

- Arduino Uno
- Capteur ultrason HC-SR04
- Driver moteur L298N
- 2 moteurs DC
- Servomoteur SG90
- Châssis 2 roues motrices
- Roue folle (roue libre)
- Batterie 7.4V (2 cellules) ou 9V avec régulateur
- Fils de connexion

FONCTIONNEMENT

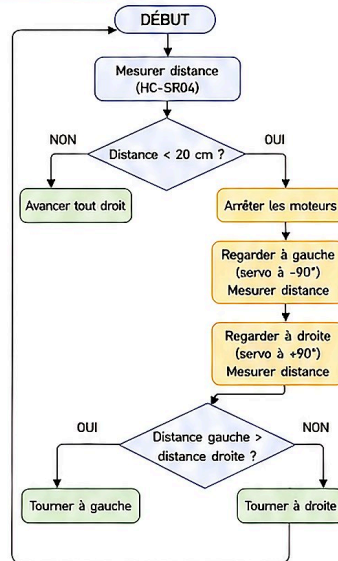
- 1 Le capteur mesure la distance.
- 2 Si la voie est libre → le robot avance.
- 3 Sinon → il s'arrête et regarde à gauche et à droite avec le servo.
- 4 Il tourne du côté où la distance est la plus grande.

SCHEMA DE CÂBLAGE



- i** Alimentez les moteurs via la borne VIN du L298N avec 7.4V - 9V. Reliez bien toutes les masses (GND) ensemble.

ORGANIGRAMME



CODE ARDUINO (EXTRAIT)

```

#include <Servo.h>

const int trigPin = 9;
const int echoPin = 10;
const int servoPin = 11;
const int in1 = 5; // Moteur gauche
const int in2 = 6;
const int in3 = 7; // Moteur droit
const int in4 = 8;

Servo servo;

long readDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  long duration = pulseIn(echoPin, HIGH);
  return duration * 0.034 / 2; // Distance en cm
}

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(in1, OUTPUT); pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT); pinMode(in4, OUTPUT);
  servo.attach(servoPin);
}

// ... (code complet disponible sur le site Dex Atomes)
  
```

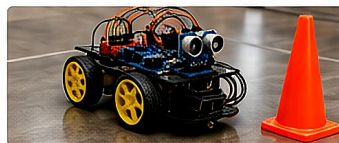
Le code complet est disponible dans les ressources téléchargeables sur le site Dex Atomes.

ASSEMBLAGE DU ROBOT

- 1 Fixez les moteurs DC de chaque côté du châssis.
- 2 Installez le driver L298N et l'Arduino au-dessus du châssis.
- 3 Fixez le capteur HC-SR04 à l'avant et le servomoteur au support.
- 4 Connectez tous les fils selon le schéma et vérifiez les connexions.

TEST ET RÉSULTATS

- ✓ Placez un obstacle à 15-20 cm devant le robot.
- ✓ Le robot doit s'arrêter automatiquement.
- ✓ Il doit regarder à gauche et à droite.
- ✓ Il doit choisir la direction la plus libre.
- ✓ Le robot continue ensuite son mouvement.



Le robot contourne l'obstacle et poursuit sa route en toute autonomie.

AMÉLIORATIONS POSSIBLES

- Ajouter un capteur infrarouge pour détecter les bordures (éviter de tomber).
- Ajouter un module Bluetooth pour contrôler le robot depuis un smartphone.
- Ajouter un écran LCD pour afficher la distance en temps réel.
- Utiliser un ESP32-CAM pour la vision robotique.
- Optimiser le code pour de meilleures performances.

À RETENIR

- ✓ Ce projet combine capteurs, actionneurs et logique de décision.
- ✓ La mesure de distance est essentielle pour la navigation autonome.
- ✓ La comparaison des distances permet d'éviter les obstacles efficacement.
- ✓ Une bonne alimentation et des connexions solides sont cruciales.
- ✓ Ce robot peut servir de base pour des projets plus avancés.



FÉLICITATIONS !

Vous venez de construire un robot autonome complet avec Arduino. Vous avez franchi une étape importante dans votre apprentissage de la robotique ! Continuez à expérimenter et à créer vos propres projets innovants.



IDÉE D'EXTENSION

Ajoutez un suiveur de ligne et transformez ce robot en robot hybride capable de suivre une ligne et d'éviter les obstacles.



BON À SAVOIR

La programmation et la robotique sont des compétences très recherchées dans l'industrie et la recherche.



CONSEIL PRATIQUE

Testez toujours chaque module séparément avant de tout assembler. Cela vous fera gagner beaucoup de temps de dépannage.

G

GLOSSAIRE DE ROBOTIQUE

Ce glossaire regroupe les principaux termes utilisés dans ce livre. Il vous aidera à mieux comprendre la robotique, l'électronique et la programmation.

GLOSSAIRE

Les définitions ci-dessous couvrent les concepts essentiels de la robotique, de l'électronique et de l'embarqué abordés tout au long de ce guide.



Actionneur :

Dispositif qui convertit une énergie (électrique, pneumatique, hydraulique) en mouvement mécanique pour effectuer une action.



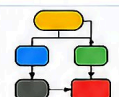
Signal :

Variation d'une grandeur physique ou électrique qui transporte de l'information (analogique ou numérique).



Arduino :

Plateforme électronique open-source basée sur un microcontrôleur facile à programmer et à utiliser pour créer des projets interactifs.



Algorithme :

Suite d'instructions logiques et ordonnées permettant de résoudre un problème ou d'accomplir une tâche.



Capteur :

Composant électronique qui détecte une grandeur physique (distance, température, lumière, etc.) et la convertit en signal électrique.



Condensateur :

Composant qui stocke temporairement des charges électriques. Utilisé pour filtrer, lisser les tensions ou temporiser.



Diode :

Composant qui laisse passer le courant dans un seul sens et le bloque dans l'autre sens.



Couple moteur :

Force de rotation produite par un moteur. Il détermine la capacité du moteur à entraîner une charge.



Transistor :

Composant utilisé comme interrupteur ou amplificateur électronique pour contrôler le courant ou la tension.



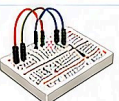
ESP32 :

Microcontrôleur puissant avec Wi-Fi et Bluetooth intégrés, idéal pour les projets IoT (Internet des objets) et la robotique avancée.



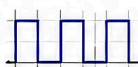
LED :

Diode électroluminescente. Émet de la lumière lorsqu'elle est parcourue par un courant dans le bon sens.



Breadboard :

Plaquette d'essai permettant de réaliser des montages électriques sans souder pour prototyper rapidement.



PWM (Modulation de largeur d'impulsion) :

Technique qui consiste à varier la largeur des impulsions pour contrôler la puissance moyenne envoyée à un actionneur (moteur, LED, etc.).



Moteur DC :

Moteur à courant continu qui tourne dans un seul sens lorsque la polarité est appliquée. Sa vitesse peut être contrôlée.



I²C :

Protocole de communication série à deux fils (SDA, SCL) permettant de connecter plusieurs périphériques avec peu de broches.



Servomoteur :

Moteur qui peut se positionner à un angle précis et le maintenir. Contrôlé par un signal PWM.



SPI :

Protocole de communication série rapide utilisant plusieurs fils (MOSI, MISO, SCK, CS) pour échanger des données.



Cinématique :

Étude du mouvement des robots sans tenir compte des forces qui causent ce mouvement.



Alimentation :

Source d'énergie électrique nécessaire au fonctionnement des composants. Peut être une batterie, une alimentation 5V, 7.4V, 9V, etc.



Repère de coordonnées :

Système de référence (X, Y, Z) utilisé pour décrire la position et l'orientation des objets et des robots dans l'espace.



Masse (GND) :

Référence zéro volt du circuit. Toutes les mesures de tension sont faites par rapport à la masse.



CONSEIL DU DÉBUTANT

Prenez le temps de relire ce glossaire chaque fois que vous rencontrez un nouveau terme. Comprendre le vocabulaire est la clé pour progresser rapidement en robotique !



À RETENIR

La robotique combine plusieurs domaines : mécanique, électronique, programmation et contrôle. Maîtriser le vocabulaire vous aidera à mieux comprendre et concevoir vos projets.



BON À SAVOIR

Les technologies évoluent rapidement. N'hésitez pas à rechercher de nouvelles informations et à expérimenter pour approfondir vos connaissances.



CONSEIL PRATIQUE

Gardez ce glossaire à portée de main pendant vos projets. Il sera votre meilleur allié pour comprendre les schémas, le code et le comportement des composants.

R

RESSOURCES ET RÉFÉRENCES UTILES



Voici une sélection de ressources en ligne, de plateformes et de livres recommandés pour approfondir vos connaissances en robotique, électronique et programmation.

POUR ALLER PLUS LOIN

La robotique est un domaine en constante évolution. Ces ressources vous aideront à continuer d'apprendre, à vous inspirer et à réaliser des projets encore plus ambitieux. Explorez, expérimentez et n'hésitez jamais à poser des questions à la communauté !



PLATEFORMES D'APPRENTISSAGE



Arduino Education
<https://education.arduino.cc>
Cours officiels, projets guidés et ressources.



Coursera
<https://www.coursera.org>
Cours en ligne sur la robotique, l'électronique et l'IA.



edX
<https://www.edx.org>
Formations universitaires en ingénierie, robotique et programmation.



Khan Academy
<https://www.khanacademy.org>
Cours gratuits de mathématiques, physique et informatique.



COMMUNAUTÉS ET FORUMS



Arduino Forum
<https://forum.arduino.cc>
Communauté officielle pour poser des questions et partager vos projets.



Stack Exchange (Robotics)
<https://robotics.stackexchange.com>
Questions techniques et discussions avancées.



Reddit – r/robotics
<https://www.reddit.com/r/robotics>
Actualités, projets et discussions sur la robotique.



Discord – Robotics
<https://discord.com>
Serveurs Discord actifs sur la robotique et l'électronique.



RÉFÉRENCES POUR COMPOSANTS



Arduino Store
<https://store.arduino.cc>
Cartes, capteurs, kits et accessoires officiels.



Adafruit
<https://www.adafruit.com>
Vaste catalogue de composants et tutoriels détaillés.

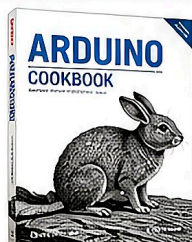


SparkFun Electronics
<https://www.sparkfun.com>
Composants électroniques et guides pour débutants.



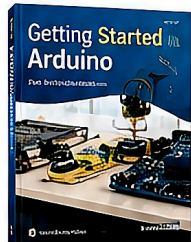
Digi-Key Electronics
<https://www.digikey.com>
Distributeur mondial de composants électroniques.

LIVRES RECOMMANDÉS



Arduino Cookbook
par Michael Margolis

Recettes et exemples pratiques pour maîtriser Arduino.



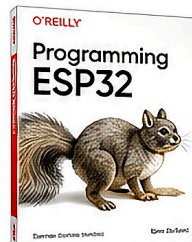
Getting Started with Arduino
par Massimo Banzi

Guide officiel pour bien démarrer avec Arduino.



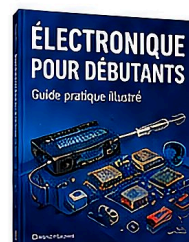
La programmation avec Arduino
par Simon Monk

Approche complète de la programmation et des projets.



Programming ESP32
par Erick Da Costa

Tout ce qu'il faut savoir pour développer avec l'ESP32.



Électronique pour débutants
par Charles Platt

Une excellente introduction à l'électronique.



LOGICIELS ET OUTILS UTILES



Arduino IDE
<https://www.arduino.cc/en/software>
Environnement de développement pour Arduino.



PlatformIO IDE
<https://platformio.org>
IDE puissant pour Arduino, ESP32 et plus.



Fritzing
<https://fritzing.org>
Pour créer des schémas électroniques.



Tinkercad Circuits
<https://www.tinkercad.com>
Simulateur en ligne pour tester vos montages.



Visual Studio Code
<https://code.visualstudio.com>
Éditeur de code léger et extensible.



CHAÎNES YOUTUBE INSPIRANTES



GreatScott!
<https://www.youtube.com/@GreatScottLab>
Électronique, débogage et projets avancés.



DroneBot Workshop
<https://www.youtube.com/@dronebotworkshop>
Tutoriels Arduino, robotique et impression 3D.



Paul McWhorter
<https://www.youtube.com/@paulmcwhorter>
Projets Arduino et électronique créatifs.



Core Electronics
<https://www.youtube.com/@coreelectronics>
Explications claires sur l'électronique.



IDÉES DE PROJETS POUR LA SUITE

- ✓ Robot suiveur de ligne
- ✓ Drone autonome
- ✓ Bras robotique à 4 degrés de liberté
- ✓ Voiture télécommandée via Bluetooth
- ✓ Système domotique (maison intelligente)
- ✓ Station météo connectée
- ✓ Imprimante 3D DIY
- ✓ Robot humanoïde miniaturisé

N'ayez pas peur d'expérimenter et de combiner plusieurs technologies pour créer vos propres projets uniques !



N'OUBLIEZ PAS !

La pratique régulière est la clé du succès en robotique. Construisez, testez, améliorez et recommencez. Chaque projet vous rendra meilleur !



La créativité, c'est l'intelligence qui s'amuse.

– Albert Einstein



EXPÉRIMENTEZ

Essayez de nouvelles idées et ne craignez pas l'échec. C'est en cherchant qu'on apprend.



DOCUMENTEZ

Prenez des notes, commentez votre code et partagez vos projets. Vous aiderez les autres et vous progresserez.



PARTAGEZ

La communauté est là pour vous aider. Partagez vos connaissances et inspirez d'autres passionnés.



RESSOURCES ESP32



ESP32 est un microcontrôleur puissant et polyvalent, idéal pour les projets IoT, la robotique et les systèmes embarqués avancés. Voici les meilleures ressources pour apprendre et créer avec ESP32.

114

RESSOURCES ESP32

ESP32 combine Wi-Fi, Bluetooth et des performances élevées dans un faible coût. Il est parfait pour les projets connectés, la domotique, la robotique, la vision artificielle et bien plus encore. Voici une sélection de ressources fiables pour maîtriser ESP32 de A à Z.



1 PLATEFORMES ET SITES OFFICIELS



Espressif Systems

<https://www.espressif.com>
Site officiel du fabricant de l'ESP32.
Datasheets, actualités et ressources.



ESP32 Documentation

<https://docs.espressif.com/projects/esp-idf>
Documentation officielle complète pour ESP32 avec ESP-IDF (C/C++).



ESP32 sur GitHub

<https://github.com/espressif>
Dépôts officiels, exemples de code et bibliothèques maintenues par Espressif.



PlatformIO

<https://platformio.org>
Environnement de développement avancé pour ESP32 (VS Code, bibliothèques, outils intégrés).



Random Nerd Tutorials

<https://randomnerdtutorials.com>
Tutoriels détaillés, guides pas à pas et projets pratiques avec ESP32.



Last Minute Engineers

<https://lastminuteengineers.com>
Projets IoT, robotique, domotique et guides pratiques pour ESP32.

2 LOGICIELS ET OUTILS RECOMMANDÉS



Arduino IDE

Facile à utiliser pour débuter rapidement avec ESP32. Prend en charge le Wi-Fi et le Bluetooth.



Visual Studio Code

Éditeur puissant avec extensions PlatformIO pour un développement professionnel.



PlatformIO IDE

Gestion des bibliothèques, compilation, débogage, moniteur série et upload simplifiés.



Node-RED

Outil de programmation visuelle pour créer des applications IoT sans écrire beaucoup de code.



MQTT Explorer

Outil pour tester et visualiser les messages MQTT avec vos projets ESP32.

3 BIBLIOTHÈQUES ET FRAMEWORKS UTILES



WiFi.h / WiFiClient.h

Bibliothèques de base pour la connexion Wi-Fi et la communication réseau.



AsyncTCP / ESPAsyncWebServer

Créer des serveurs web rapides et réactifs sur ESP32.



ArduinoJson

Manipulation simple et efficace de données JSON.



FreeRTOS

Système d'exploitation temps réel pour gérer plusieurs tâches de manière efficace.



ESP32 Camera Library

Contrôler la caméra intégrée de l'ESP32 CAM pour la vision et l'IA.

4 IDÉES DE PROJETS AVEC ESP32

Station météo connectée



Mesure la température, l'humidité et envoie les données sur Internet.

Domotique intelligente



Contrôle des lumières, écrans, appareils via Wi-Fi ou smartphone.

Robot Wi-Fi télécommandé



Pilotez votre robot à distance depuis une application Web.

Caméra de surveillance



Diffuse la vidéo en direct sur le Web avec l'ESP32-CAM.

Système d'alarme IoT



Détection de mouvement et alerte envoyée sur votre téléphone.

5 CONSEILS POUR DÉBUTER AVEC ESP32

- ✓ Commencez par des exemples simples avant des projets complexes.
- ✓ Utilisez le moniteur série pour déboguer et comprendre votre code.
- ✓ Alimentez correctement votre ESP32 (3.3V recommandé).
- ✓ Mettez à jour régulièrement vos bibliothèques et l'IDE.
- ✓ Sécurisez vos projets connectés (mots de passe, HTTPS, authentification).
- ✓ Explorez les projets open-source pour apprendre des autres.

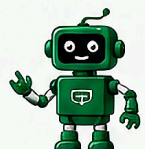


Apprendre. Créer. Innover. Partager.
C'est l'esprit ESP32 !



ASTUCE

L'ESP32 est très polyvalent. Ne vous limitez pas aux exemples : expérimentez, combinez les capteurs, les communications et l'intelligence artificielle pour créer des projets uniques et innovants !



COMMUNAUTÉ ACTIVE

Des milliers de développeurs partagent leurs projets et solutions chaque jour. Rejoignez la communauté ESP32 !



APPRENTISSAGE CONTINU

La technologie évolue vite. Continuez à apprendre et à pratiquer pour rester à jour.



CRÉEZ VOTRE AVENIR

Avec ESP32, vous avez le pouvoir de transformer vos idées en solutions réelles et utiles.

Scannez ce code pour accéder à encore plus de ressources ESP32 !



R ROBOTIQUE AVANCÉE



Pour aller plus loin dans la robotique, il est important de découvrir des technologies avancées utilisées dans l'industrie et la recherche.

115 ROBOTIQUE AVANCÉE

La robotique moderne ne se limite pas aux moteurs et aux capteurs. Elle s'appuie sur des logiciels puissants, de l'intelligence artificielle, la vision par ordinateur et des systèmes complexes pour réaliser des robots autonomes et intelligents.



1 LOGICIELS ET FRAMEWORKS DE ROBOTIQUE



ROS (Robot Operating System)

Framework open-source pour développer des applications robotiques complexes. Utilisé dans l'industrie, la recherche et l'éducation.



ROS 2

Version moderne de ROS, plus rapide, plus sûre et adaptée aux systèmes distribués.



MoveIt!

Plateforme de planification de mouvement pour les bras robotiques et les manipulateurs.



Gazebo

Simulateur 3D réaliste pour tester des robots et des environnements avant le déploiement réel.



OpenCV

Bibliothèque de vision par ordinateur pour le traitement d'images et la détection d'objets.

2 DOMAINES AVANCÉS



Vision par ordinateur

Permet aux robots de comprendre leur environnement à l'aide de caméras et d'algorithmes d'analyse d'images.



Intelligence artificielle

Apprentissage automatique et réseaux de neurones pour la perception, la décision et l'autonomie.



Planification de trajectoire

Algorithmes pour calculer le meilleur chemin d'un point à un autre en évitant les obstacles.



SLAM (Simultaneous Localization and Mapping)

Permet au robot de se localiser et de construire une carte de l'environnement en même temps.



Navigation autonome

Combinaison de capteurs, cartes et algorithmes pour déplacer un robot sans intervention humaine.

3 TYPES DE ROBOTS AVANCÉS

Robots mobiles autonomes (AMR)



Déplacement autonome dans des environnements dynamiques.

Drones intelligents



Utilisés pour la surveillance, la livraison et la cartographie.

Bras robotiques industriels



Haute précision pour l'assemblage, la soudure et la manipulation.

Robots humanoïdes



Conçus pour interagir avec les humains et leur environnement.

Robots sous-marins (ROV)



Exploration sous-marine, inspection de structures et recherche scientifique.

4 APPLICATIONS INDUSTRIELLES



Automatisation industrielle

Châînes de production, contrôle qualité, assemblage.



Médecine et chirurgie

Robots chirurgicaux, exosquelettes, assistance.



Agriculture intelligente

Robots pour la récolte, le tri et la surveillance des cultures.



Logistique et entrepôts

Robots de transport, tri automatique, gestion des stocks.



Exploration spatiale

Robots pour explorer d'autres planètes et environnements extrêmes.

5 CAPTEURS ET TECHNOLOGIES CLÉS

LIDAR



Mesure des distances par laser pour créer des cartes 3D.

Caméra 3D



Perception de profondeur et détection d'objets.

IMU



Mesure de l'accélération et l'orientation du robot.

Capteur de force



Détecte la force et le contact pour des grip précis.

GPS RTK



Positionnement précis au centimètre près pour robots mobiles.

6 TENDANCES ET INNOVATIONS

- ✓ Robotique collaborative (cobots)
- ✓ Apprentissage par renforcement
- ✓ Robots souples et bio-inspirés
- ✓ Robotique modulaire
- ✓ Intégration IA + Robotique
- ✓ Essais de robots (swarm robotics)



L'avenir de la robotique appartient à ceux qui osent apprendre, expérimenter et construire.



CONSEILS POUR ALLER PLUS LOIN



Approfondissez un sujet à la fois.



Construisez des projets réels et documentez-les.



Participez à des communautés et partagez vos expériences.



Ne craignez pas l'échec : c'est en testant qu'on progresse !



Ressources en ligne

De nombreux cours, tutoriels et documentations sont disponibles gratuitement. Profitez-en !



Communauté

Rejoignez des forums, groupes et événements pour rencontrer d'autres passionnés.



Formation continue

La technologie évolue vite. Restez curieux et continuez d'apprendre toute votre vie.

Scannez ce QR code pour accéder à une liste complète de ressources en robotique.



R

RESSOURCES ROBOTIQUE



Voici une sélection des meilleures ressources pour découvrir, apprendre et approfondir tous les aspects de la robotique moderne.

116

RESSOURCES ROBOTIQUE

La robotique est un domaine vaste et passionnant qui regroupe mécanique, électronique, programmation, IA et conception. Ces ressources vous aideront à aller plus loin et à devenir un meilleur roboticien.

1 PLATEFORMES D'APPRENTISSAGE EN LIGNE



edX

<https://www.edx.org>
Cours universitaires en robotique, IA, contrôle et automatisation par des universités de renommée mondiale.



Coursera

<https://www.coursera.org>
Spécialisations et cours sur la robotique, ROS, IA et vision par ordinateur.



Udemy

<https://www.udemy.com>
Formations pratiques pour tous les niveaux : Arduino, ROS, drones, impression 3D, etc.



FutureLearn

<https://www.futurelearn.com>
Cours en ligne avec des projets guidés par des experts internationaux.



2 COMMUNAUTÉS ET FORUMS



ROS Discourse

<https://discourse.ros.org>
Forum officiel de la communauté ROS pour poser des questions et partager des solutions.



r/robotics (Reddit)

<https://www.reddit.com/r/robotics>
Communauté active pour discuter de robotique, projets, actu et innovations.



Robotics Stack Exchange

<https://robotics.stackexchange.com>
Questions techniques et réponses détaillées sur la robotique et l'électronique.



IEEE Robotics & Automation

<https://www.ieee-ras.org>
Ressources, articles scientifiques et événements autour de la robotique avancée.

3 LIVRES RECOMMANDÉS



Robot Modeling and Control

par Mark W. Spong
Référence complète sur la modélisation et le contrôle des robots.



Introduction to Robotics: Mechanics and Control

par John J. Craig
Idéal pour comprendre la mécanique et les bases du contrôle.



Programming Robots with ROS

par Morgan Quigley
Guide pratique pour apprendre ROS avec des projets concrets.



Artificial Intelligence: A Modern Approach

par Stuart Russell & Peter Norvig
Comprendre l'IA pour créer des robots intelligents et autonomes.

4 CHAÎNES YOUTUBE UTILES



The Construct

Chaîne dédiée à la robotique, aux projets DIY, aux robots humanoïdes et à l'IA.



Add0hms

Projets électroniques, Arduino, impression 3D et robotique créative.



ROS Developers

Tutoriels officiels ROS, démonstrations et présentations techniques.



GreatScott!

Électronique, débogage et projets avancés en robotique.



James Bruton

Conception et construction de robots, tests et ingénierie pratique.

5 LOGICIELS ET OUTILS INDISPENSABLES



ROS (Robot Operating System)

Framework open-source pour développer des robots complexes.



Gazebo

Simulateur 3D pour tester des robots dans des environnements réalistes.



Unity

Moteur 3D puissant pour la simulation et la visualisation robotique.



OpenCV

Bibliothèque de vision par ordinateur pour la détection et le traitement d'images.



SolidWorks / Fusion 360

Logiciels de CAO pour concevoir les pièces et structures de vos robots.

6 DOMAINES DE LA ROBOTIQUE À EXPLORER

Robotique mobile



Déplacement autonome, navigation et évitement d'obstacles.

Robotique industrielle



Automatisation des tâches, bras robotisés et chaînes de production.

Robotique humanoïde



Conception de robots qui interagissent avec les humains.

Drone et aéronautique



Drones autonomes, contrôle de vol et applications aériennes.

Robotique sous-marine



Exploration marine, robots ROV et études sous-marines.

Bio-robotique



Interaction avec le corps humain et technologies médicales.

IA et robotique



Intelligence artificielle appliquée à la perception, à la décision et à l'autonomie.



CONSEIL

La pratique est la clé ! Commencez par de petits projets, participez aux communautés et n'ayez pas peur d'échouer. Chaque robot est une leçon précieuse.



RAPPEL IMPORTANT

La sécurité avant tout : testez toujours vos robots dans un environnement contrôlé et respectez les règles de sécurité électrique et mécanique.



EXPLOREZ

Le monde de la robotique évolue chaque jour. Restez curieux et continuez d'apprendre !



PARTAGEZ

Partagez vos projets, posez des questions et inspirez les autres membres de la communauté.



PROGRESSEZ

Du débutant à l'expert, chaque étape vous rapproche de la création de robots incroyables.

Scannez ce QR code pour accéder à une liste complète de ressources robotique !



R PROJETS À RÉALISER APRÈS CE LIVRE



La meilleure façon de progresser en robotique est de pratiquer ! Voici des idées de projets classés par niveau pour continuer à apprendre, créer et innover.

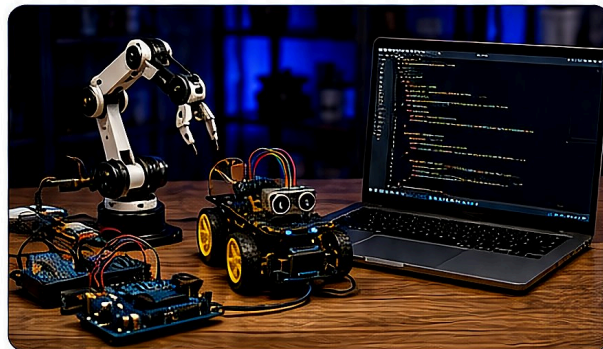
117

PROJETS À RÉALISER APRÈS CE LIVRE

La théorie est importante, mais la pratique est essentielle !

Ces projets vous permettront de consolider vos connaissances, de relever des défis concrets et de développer votre créativité.

Choisissez un projet selon votre niveau et lancez-vous !



1 NIVEAU DÉBUTANT

Idéal pour consolider les bases



Clignotant LED

Faites clignoter une LED avec Arduino.



Feu tricolore

Simulez un feu de circulation avec trois LEDs.



Thermomètre numérique

Affichez la température mesurée par un capteur sur l'écran OLED.



Suiveur de ligne basique

Robot à 2 roues qui suit une ligne noire.



Ultrason : mesure de distance

Mesurez la distance avec un capteur ultrason HC-SR04 et affichez-la.

2 NIVEAU INTERMÉDIAIRE

Pour aller plus loin



Robot évitant les obstacles

Utilisez un ultrason pour détecter les obstacles et évitez-les.



Bras robotique 3D

Contrôlez un bras robotique avec des servomoteurs.



Station météo

Mesurez température, humidité et pression, puis affichez les données.



Voiture télécommandée

Contrôlez une voiture à distance via Bluetooth.



Serrure électronique

Ouvrez une serrure avec un clavier et un servo.

3 NIVEAU AVANCÉ

Pour les passionnés et créateurs



Robot autonome avec caméra

Utilisez la vision par ordinateur pour détecter et suivre des objets.



Drone contrôlé par smartphone

Créez ou programmez un drone et pilotez-le depuis votre téléphone.



ROV sous-marin

Concevez et contrôlez un robot sous-marin avec caméra.



Robot humanoïde mini

Imprimez, assemblez et programmez un petit robot humanoïde.



Projet IA embarquée

Ajoutez une intelligence artificielle (classification d'images, suivi, reconnaissance) à votre robot.

4 COMMENT RÉUSSIR VOS PROJETS



Planifiez votre projet

Définissez un objectif clair et listez les composants nécessaires.



Comprenez chaque étape

Prenez le temps de comprendre le code et le fonctionnement.



Testez et améliorez

Testez, corrigez, puis améliorez votre projet régulièrement.



Partagez votre travail

Montrez votre projet à la communauté et demandez des avis.



Documentez

Gardez des notes, schémas et photos de vos projets.

5 MÉTHODOLOGIE DE PROJET



1

IDÉE

Choisissez un projet qui vous motive.



2

PLANIFICATION

Dessinez, listez les composants et définissez le plan.



3

DÉVELOPPEMENT

Écrivez le code, assemblez et testez.



4

TEST & DÉBOGAGE

Corrigez les erreurs et optimisez.



5

FINALISATION

Améliorez le design et les performances.



6

PARTAGE

Publiez votre projet et inspirez les autres !

6 RESSOURCES UTILES POUR VOS PROJETS



YouTube

Tutoriels vidéo pour apprendre en pratiquant.

<https://www.youtube.com>



GitHub

Code source, bibliothèques et projets open-source.

<https://github.com>



Instructables

Guides de projets étape par étape.

<https://www.instructables.com>



Thingiverse

Modèles 3D gratuits pour l'impression.

<https://www.thingiverse.com>



Tinkercad Circuits

Simulez vos circuits en ligne avant de construire.

<https://www.tinkercad.com>



Hackaday

Idées, articles techniques et inspirations.

<https://hackaday.io>



CHAQUE PROJET EST UNE VICTOIRE !

Ne cherchez pas la perfection, cherchez le progrès. Expérimentez, échouez, apprenez et recommencez. C'est ainsi que naissent les vrais ingénieurs !



REJOIGNEZ LA COMMUNAUTÉ

Échangez, posez vos questions et partagez vos projets avec des milliers de passionnés de robotique.



CONTINUEZ À APPRENDRE

La robotique évolue chaque jour. Restez curieux, apprenez sans cesse et construisez votre avenir.



INNOVEZ ET CRÉEZ

Utilisez vos compétences pour résoudre des problèmes réels et créer les robots de demain !

Scannez-moi pour découvrir encore plus de ressources et d'idées de projets !



R SÉCURITÉ ET BONNES PRATIQUES

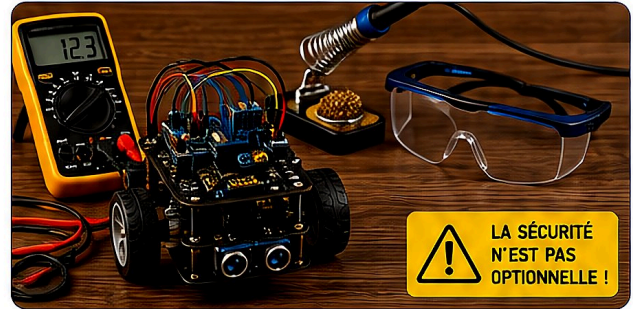


La sécurité doit toujours être votre priorité lorsque vous concevez, construisez et utilisez des robots ou des systèmes électroniques. Ces règles simples vous aideront à protéger votre matériel et, surtout, vous-même.

118

SÉCURITÉ ET BONNES PRATIQUES

Travailler avec l'électronique, la mécanique et la robotique implique des risques. En suivant ces bonnes pratiques, vous éviterez les accidents, prolongerez la durée de vie de vos composants et assurerez le bon fonctionnement de vos projets.



1 SÉCURITÉ ÉLECTRIQUE



Vérifiez toujours la tension et le courant

Assurez-vous que vos composants supportent la tension et le courant utilisés.



Coupez l'alimentation avant de connecter

Toujours débrancher ou éteindre avant de faire un branchement.



Utilisez des alimentations adaptées

Ne jamais utiliser une alimentation avec une tension plus élevée que nécessaire.



Protégez vos circuits

Utilisez des fusibles ou des protections contre les surintensités.



Attention aux décharges électrostatiques (ESD)

Utilisez un bracelet antistatique pour manipuler les composants sensibles.

2 SÉCURITÉ MÉCANIQUE



Protégez les parties mobiles

Utilisez des carters, des couvercles ou des grilles pour éviter tout contact accidentel avec les engrenages, les courroies ou les bras robotiques.



Gardez vos mains éloignées

Ne jamais mettre vos doigts près des moteurs ou des mécanismes en mouvement.



Serrez correctement les vis et fixations

Des pièces mal fixées peuvent se détacher et provoquer des accidents.



Stabilité et équilibre

Assurez-vous que votre robot est stable et ne risque pas de basculer ou de tomber.



Ne surchargez pas votre robot

Respectez les limites de charge des moteurs et des structures.

3 SÉCURITÉ LORS DES TESTS ET PROGRAMMATION



Testez étape par étape

Testez chaque module séparément avant d'assembler le système complet.



Vérifiez votre code

Une erreur de programmation peut provoquer des mouvements inattendus.



Surveillez les premiers tests

Restez toujours près de votre robot lors des premiers essais.



Surveillez la batterie

Une batterie endommagée ou gonflée peut être dangereuse.



Évitez la surchauffe

Vérifiez que les moteurs et les composants ne chauffent pas trop.

ÉQUIPEMENTS DE PROTECTION RECOMMANDÉS



Lunettes de sécurité



Gants isolants



Bracelet antistatique



Extracteur de fumées

EPI = Équipement de Protection Individuelle

Investir dans la sécurité aujourd'hui vous évitera des problèmes demain.

4 BONNES PRATIQUES GÉNÉRALES

- ✓ Travaillez dans un espace propre et bien éclairé.
- ✓ Rangez vos outils et composants après chaque session.
- ✓ Lisez les datasheets de vos composants.
- ✓ Prenez des notes et documentez vos projets.
- ✓ Sauvegardez régulièrement votre code.
- ✓ Apprenez de vos erreurs et soyez patient !



5 GESTION DES CÂBLES

- ✓ Utilisez des attaches câbles pour organiser vos fils.
- ✓ Évitez que les câbles touchent les parties mobiles.
- ✓ Identifiez vos câbles (étiquettes ou couleurs).
- ✓ Évitez les câbles trop longs qui peuvent s'emmêler.



6 STOCKAGE ET ENTRETIEN

- ✓ Gardez vos composants à l'abri de l'humidité et de la poussière.
- ✓ Vérifiez régulièrement l'état de vos batteries et câbles.
- ✓ Nettoyez vos robots après chaque utilisation.
- ✓ Remplacez les pièces usées ou endommagées à temps.



RAPPELEZ-VOUS

Un bon robot est le résultat d'un bon design, mais un robot sûr est le résultat de bonnes habitudes.



VOTRE SÉCURITÉ D'ABORD

Vous êtes l'ingénieur le plus important de votre projet. Prenez soin de vous !



APPRENDRE EN TOUTE SÉCURITÉ

Respectez les règles, suivez les bonnes pratiques et vous progresserez plus loin.

Scannez ce QR code pour découvrir plus de conseils de sécurité en robotique.



R DÉPANNAGE ET RÉOLUTION DE PROBLÈMES

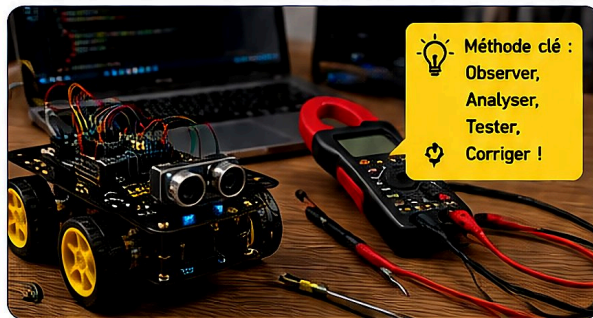


Les problèmes font partie de tout projet de robotique. L'important est de savoir les identifier, les comprendre et les résoudre méthodiquement. Ce chapitre vous aide à gagner du temps et à éviter la frustration !

119

DÉPANNAGE ET RÉSOLUTION DE PROBLÈMES

Que votre robot ne bouge pas, que vos capteurs donnent des valeurs étranges ou que votre code ne fonctionne pas comme prévu, cette section vous guide pas à pas pour trouver la cause et corriger le problème.



Méthode clé :
Observer,
Analyser,
Tester,
Corriger !

1 PROBLÈMES COURANTS ET SOLUTIONS RAPIDES

Problème	Causes possibles	Solutions	À vérifier
Le robot ne s'allume pas	- Batterie déchargée - Mauvaise connexion - Interrupteur défectueux	- Charger la batterie - Vérifier toutes les connexions - Tester ou remplacer l'interrupteur	Tension de la batterie, fusibles, câbles.
Les moteurs ne tournent pas	- Câblage incorrect - Driver moteur mal connecté - Code ou commande incorrecte	- Vérifier les branchements - Examiner le driver (L298N, etc.) - Tester avec un code simple	Polarité des moteurs, alimentation du driver, signaux du code.
Lectures de capteur incorrectes	- Mauvaise connexion - Capteur mal placé - Bruit électrique	- Vérifier les câbles - Repositionner le capteur - Ajouter filtrage ou moyenne	Alimentation du capteur, distance, environnement.
Le code ne se téléverse pas	- Mauvais port COM - Carte non reconnue - Pilote manquant	- Sélectionner le bon port - Vérifier le câble USB - Installer les pilotes nécessaires	Port dans l'IDE, câble USB, drivers.
Bluetooth / Wi-Fi ne fonctionne pas	- Mauvais appairage - Interférences - Mauvaise configuration	- Appairer à nouveau - Réduire les interférences - Vérifier la configuration du code	Paramètres de module, antenne, alimentation.

2 MÉTHODE DE DÉPANNAGE EN 5 ÉTAPES



1 OBSERVER

Décrivez le problème précisément. Que se passe-t-il exactement ?



2 ANALYSER

Quelles sont les causes possibles ? Vérifiez le schéma et le code.



3 TESTER

Testez chaque cause possible une par une.



4 CORRIGER

Appliquez la solution identifiée.



5 VÉRIFIER

Testez à nouveau pour confirmer que le problème est résolu.

3 OUTILS UTILES POUR LE DÉPANNAGE



Multimètre
Mesure la tension, le courant et la résistance.



Oscilloscope (si disponible)
Visualise les signaux électriques en temps réel.



Câble USB de qualité
Évite les erreurs de connexion et de transfert.



Breadboard
Permet de tester rapidement un circuit sans soudure.



Moniteur série
Affiche les valeurs et messages pour comprendre le comportement.

4 CONSEILS POUR GAGNER DU TEMPS

- ✓ Commencez simple : testez chaque partie séparément avant d'assembler le tout.
- ✓ Gardez votre espace de travail propre et bien organisé.
- ✓ Commenter votre code pour comprendre vos actions.
- ✓ Prenez des notes de vos tests et résultats.
- ✓ Sauvegardez souvent votre code.
- ✓ Utilisez des noms clairs pour vos variables et fonctions.
- ✓ Ne changez qu'une chose à la fois lors du test.
- ✓ Demandez de l'aide à la communauté si besoin (forums, Discord, etc.).



5 EXEMPLES DE VÉRIFICATIONS RAPIDES

Alimentation



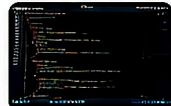
- ✓ Batterie chargée ?
- ✓ Interrupteur ON ?
- ✓ Câbles bien connectés ?
- ✓ Tension suffisante ?

Connexions



- ✓ Fils bien insérés ?
- ✓ Polarité respectée ?
- ✓ Soudures solides ?
- ✓ Pas de câble lâche ?

Code



- ✓ Bon port sélectionné ?
- ✓ Pas d'erreur de compilation ?
- ✓ Bon type de carte ?
- ✓ Bibliothèques installées ?

Capteurs



- ✓ Alimentation correcte ?
- ✓ Câblage correct ?
- ✓ Distance / angle OK ?
- ✓ Pas d'obstacle parasite ?

Moteurs



- ✓ Driver alimenté ?
- ✓ Câbles moteurs OK ?
- ✓ Signaux PWM corrects ?
- ✓ Code commande OK ?

6 QUAND TOUT SEMBLE BLOQUÉ...



Faites une pause !

Reprenez calmement les étapes, relisez votre schéma et votre code. Parfois, un regard neuf aide à trouver la solution.



La persévérance est la clé de tout roboticien !



RAPPELEZ-VOUS

Chaque problème résolu vous rend plus fort ! Le dépannage est une compétence essentielle en robotique. Plus vous pratiquez, plus vous devenez rapide et efficace.



COMMUNAUTÉ

Rejoignez des groupes et forums pour partager vos problèmes et solutions.



DOCUMENTATION

Consultez toujours la documentation officielle des composants et bibliothèques.



APPROFONDISSEZ

Chaque problème est une occasion d'apprendre quelque chose de nouveau.

Scannez ce QR code pour accéder à une liste de checklists et guides de dépannage téléchargeables.



R

QUESTIONS FRÉQUENTES (FAQ)

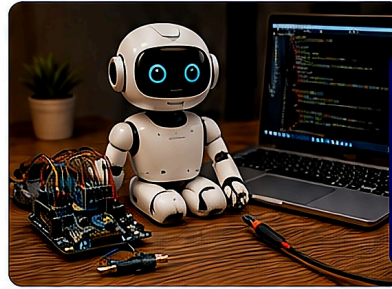


Voici les réponses aux questions les plus courantes que se posent les débutants en robotique, Arduino et électronique. Si vous débutez, ne vous inquiétez pas : chaque expert était un débutant un jour !

120

QUESTIONS FRÉQUENTES (FAQ)

Cette section répond aux questions que vous vous posez peut-être. Si votre question ne figure pas ici, rejoignez notre communauté et posez-la !



ASTUCE

La curiosité est votre meilleur outil en robotique. Posez des questions, testez et pratiquez chaque jour !

1 Ai-je besoin de connaissances en électronique pour commencer la robotique ?



Non ! Ce livre est conçu pour les débutants. Nous expliquons tout depuis les bases avec des exemples simples et pratiques.

2 Quelle est la différence entre Arduino et ESP32 ?



Arduino est plus simple et idéal pour les projets basiques. L'ESP32 est plus puissant, possède le Wi-Fi et le Bluetooth, et convient aux projets IoT et avancés.

3 Quel langage de programmation est utilisé ?



Nous utilisons le langage C/C++ simplifié avec l'IDE Arduino. Si vous savez déjà programmer, vous serez à l'aise rapidement.

4 Où acheter les composants et modules ?



Vous pouvez acheter sur Amazon, AliExpress, eBay ou dans les magasins d'électronique locaux. Les liens recommandés sont dans le chapitre « Ressources ».

5 Est-ce que je peux réaliser les projets sans imprimante 3D ?



Oui, la plupart des projets peuvent être réalisés sans impression 3D. Vous pouvez utiliser des châssis disponibles dans le commerce ou des matériaux simples.

6 Combien coûte un kit de démarrage en robotique ?



Un bon kit de démarrage coûte entre 50 € et 120 € selon la qualité et le nombre de composants. Commencez petit, puis améliorez progressivement.

7 Quelle alimentation dois-je utiliser ?



Pour la plupart des projets Arduino, une alimentation 5V (USB ou batterie) suffit. Pour les moteurs, utilisez une alimentation séparée adaptée au courant requis.

8 Comment éviter d'endommager mes composants ?



Vérifiez toujours les connexions avant d'alimenter, respectez les tensions et courants maximums, et utilisez des résistances pour les LED.

9 Pourquoi mon code ne fonctionne pas ?



Les causes les plus courantes : erreur de câblage, mauvais port sélectionné, bibliothèque manquante ou erreur de syntaxe. Vérifiez étape par étape.

10 Comment apprendre plus rapidement la robotique ?



Pratiquez régulièrement, construisez des projets, posez des questions, et n'ayez pas peur d'échouer ! Chaque erreur est une leçon.

11 Puis-je travailler sur les projets du livre si je suis débutant complet ?



Oui ! Les projets sont classés par niveau : débutant, intermédiaire et avancé. Commencez par les projets débutants et progressez à votre rythme.

12 Où trouver de l'aide si je bloque ?



Rejoignez les communautés (Discord, Facebook, Reddit), consultez les forums (Arduino Forum) ou regardez les vidéos recommandées dans le livre.

13 QUELS SONT LES TYPES DE ROBOTS QUE JE PEUX CRÉER ?

Robots mobiles



Se déplacent autonomement et peuvent éviter les obstacles.

Bras robotique



Effectuent des mouvements précis pour saisir et manipuler des objets.

Robots télécommandés



Contrôlés à distance via Bluetooth, Wi-Fi ou télécommande.

Robots IA / Vision



Utilisent des caméras et l'IA pour reconnaître leur environnement.

14 COMBIEN DE TEMPS FAUT-IL POUR DEVENIR BON EN ROBOTIQUE ?

Cela dépend de votre engagement, mais avec 30 minutes à 1 heure par jour, vous pouvez :

- ✓ Comprendre les bases en 2 à 4 semaines
- ✓ Réaliser des projets simples en 1 à 2 mois
- ✓ Construire des robots avancés en 6 à 12 mois



La clé est la régularité et la pratique ! La patience et la persévérance paient toujours.

CONSEILS RAPIDES



Fixez-vous des objectifs réalistes et progressez étape par étape.



Prenez des notes de vos tests et idées pour ne rien oublier.



Regardez des tutoriels et inspirez-vous des projets des autres.



Expérimentez, modifiez et améliorez vos projets sans crainte.



Amusez-vous ! La robotique doit être créative et passionnante.



BESOIN D'AIDE ?

Rejoignez notre communauté DexLab pour poser vos questions et partager vos projets avec d'autres passionnés !



RESSOURCES UTILES

Tous les liens, forums, bibliothèques et tutoriels sont disponibles dans le chapitre « Pour aller plus loin ».

SCANNEZ MOI

Accédez à notre serveur Discord et à une liste de ressources exclusives pour vous aider !



R RÉSUMÉ DU LIVRE



Félicitations !
Vous avez parcouru tout ce livre et acquis les connaissances essentielles pour comprendre, créer et programmer des robots modernes. Ce résumé vous aide à revoir les points clés et à consolider vos apprentissages.

121

RÉSUMÉ DU LIVRE

Ce livre vous a guidé pas à pas dans l'univers de la robotique, de ses bases théoriques à la réalisation de projets concrets.

Voici un récapitulatif des connaissances et compétences que vous avez acquises.



“ La robotique n'est pas seulement une technologie, c'est une façon de résoudre des problèmes et d'améliorer le monde. ”

★ 1 CE QUE VOUS AVEZ APPRIS DANS CE LIVRE



1. ROBOTIQUE

Vous avez découvert l'histoire, les types de robots, leurs composants, la cinématique, la dynamique, les capteurs, les actionneurs et les bases du contrôle.



2. ÉLECTRONIQUE

Vous avez appris les principes fondamentaux de l'électronique : lois des circuits, composants passifs et actifs, filtres, alimentations et réalisation de circuits.



3. ARDUINO ET ESP32

Vous savez maintenant programmer Arduino et ESP32, utiliser leurs entrées/sorties, communiquer, afficher des données et connecter divers capteurs et modules.



4. CAPTEURS ET ACTIONNEURS

Vous comprenez le fonctionnement des capteurs courants et des actionneurs, et vous savez les intégrer dans vos projets pour interagir avec le monde réel.



5. PROJETS PRATIQUES

Vous avez réalisé de nombreux projets : robots mobiles, bras robotique, robots suiveurs de ligne, systèmes à distance et stations météo, etc.



6. SÉCURITÉ ET DÉPANNAGE

Vous connaissez les bonnes pratiques de sécurité, les méthodes de test, de dépannage et de résolution de problèmes pour construire des systèmes fiables.



COMPÉTENCES ACQUISES

- ✓ Analyser un problème et proposer une solution robotique.
- ✓ Concevoir et câbler des circuits électroniques.
- ✓ Programmer des cartes Arduino et ESP32.
- ✓ Utiliser des capteurs et actionneurs efficacement.
- ✓ Communiquer entre différents modules et composants.
- ✓ Construire et tester des robots fonctionnels.
- ✓ Optimiser et améliorer un projet existant.
- ✓ Travailler en autonomie et résoudre des problèmes.



OUTILS ET TECHNOLOGIES MAÎTRISÉS



Arduino IDE



ESP32 / ESP-IDF / PlatformIO



Capteurs : ultrasons, IR, DHT11, MPU6050, caméra, etc.



Afficheurs : LCD, OLED, 7 segments



Communication : I2C, SPI, UART, Bluetooth, Wi-Fi



Outils : Multimètre, Oscilloscope, Imprimante 3D, Logiciels CAO



EXEMPLES DE PROJETS RÉALISÉS

- Robot suiveur de ligne
- Bras robotique 3D contrôlé
- Robot évitant les obstacles
- Station météo connectée
- Voiture télécommandée
- Robot contrôlé par smartphone



ET MAINTENANT ?



Ce livre n'est que le début de votre aventure en robotique ! Continuez à apprendre, à expérimenter et à créer. Voici quelques idées pour aller plus loin :



Approfondir la programmation (C++, MicroPython, ROS, etc.)



Explorer l'IA et la vision par ordinateur pour des robots intelligents



Développer des projets IoT et connectés avec le cloud



Concevoir et imprimer vos propres pièces en 3D



Participer à des concours et projets open source



Partager vos projets et aider la communauté



RAPPEL IMPORTANT

La persévérance est la clé du succès en robotique. N'ayez pas peur d'échouer, chaque erreur est une occasion d'apprendre et de progresser.



MERCI !

Merci d'avoir choisi ce livre pour votre apprentissage. Continuez à rêver, à construire et à innover. Le futur appartient à ceux qui créent !



REJOIGNEZ LA COMMUNAUTÉ

Échangez vos idées, posez vos questions et partagez vos projets avec d'autres passionnés de robotique.



CONTINUEZ À APPRENDRE

Ne cessez jamais d'apprendre. La technologie évolue chaque jour, restez curieux et motivé.



RÉALISEZ VOS IDÉES

Vos idées peuvent changer le monde. Transformez vos connaissances en solutions concrètes.

SCANNEZ CE QR CODE

Accédez à la liste complète des ressources, projets et guides supplémentaires.



R

CONCLUSION ET MOT DE LA FIN



Vous êtes arrivé au bout de ce voyage passionnant dans l'univers de la robotique !

Continuez à explorer, à apprendre et surtout, à créer.

Le meilleur robot que vous construirez sera toujours le prochain !

122

CONCLUSION ET MOT DE LA FIN

La robotique est un domaine vaste et en constante évolution. Ce livre vous a fourni les bases essentielles, les outils et la méthodologie pour comprendre, concevoir et réaliser des robots.

Mais l'aventure ne fait que commencer !

Le futur de la robotique se construit avec des esprits curieux, créatifs et déterminés comme le vôtre.



“ La technologie change le monde, mais ce sont les ingénieurs et les créateurs qui le rendent meilleur.

CE QUE VOUS POUVEZ FAIRE MAINTENANT



1. CONTINUEZ À APPRENDRE

- Ne vous arrêtez jamais. La curiosité est votre meilleur moteur.
- Approfondissez les sujets qui vous passionnent.



2. PRATIQUEZ RÉGULIÈREMENT

- La pratique transforme les connaissances en compétences.
- Réalisez des projets et relevez des défis concrets.



3. PARTAGEZ ET COLLABOREZ

- Échangez vos idées, vos projets et vos solutions avec la communauté.
- Ensemble, nous allons plus loin !



4. INNOVEZ ET CRÉEZ

- Imaginez de nouvelles solutions pour résoudre des problèmes réels.
- Votre prochaine idée peut avoir un impact réel sur le monde.



5. PERSÉVÉREZ

- Chaque erreur est une leçon.
- La persévérance fait la différence entre un projet abandonné et une réussite.

RÉCAPITULATIF DU LIVRE

- ✓ Chapitre 1 : Introduction à la robotique
- ✓ Chapitre 2 : Électronique pour la robotique
- ✓ Chapitre 3 : Arduino et ESP32
- ✓ Chapitre 4 : Capteurs et actionneurs
- ✓ Chapitre 5 : Programmation et contrôle
- ✓ Chapitre 6 : Projets pratiques
- ✓ Chapitre 7 : Aller plus loin et ressources



VOTRE BOÎTE À OUTILS DE DÉPART



Arduino / ESP32



Breadboard



Multimètre



Capteurs



Actionneurs



IDE & Code



Outils



Documentation



Curiosité

DOMAINES D'APPLICATION DE LA ROBOTIQUE



Industrie : automatisation, bras robotisés, contrôle qualité.



Médecine : chirurgie assistée, rééducation, prothèses intelligentes.



Agriculture : robots autonomes, drones, surveillance des cultures.



Exploration : robots spatiaux, sous-marins, missions de reconnaissance.



Services : robots domestiques, livraison, assistance aux personnes.



Éducation : apprentissage, recherche, découverte et innovation.



Rappelez-vous :

La robotique n'est pas seulement une compétence technique, c'est une façon de penser, de résoudre et d'améliorer le monde.

VOUS FAITES PARTIE DE CETTE RÉVOLUTION !

VOS PROCHAINS DÉFIS

- ✓ Construire un robot autonome complet
- ✓ Créer un système IoT connecté
- ✓ Développer une application mobile pour contrôler un robot
- ✓ Utiliser l'intelligence artificielle dans vos projets
- ✓ Participer à un concours ou à un hackathon
- ✓ Enseigner et inspirer d'autres personnes



♥ MERCI !

Merci d'avoir choisi ce livre pour votre apprentissage. J'espère qu'il vous a donné les connaissances, la confiance et l'inspiration pour réaliser vos propres projets et construire le futur.

Continuez à rêver, à apprendre et à créer ! Le monde a besoin de vos idées.



RESSOURCES SUPPLÉMENTAIRES



Sites officiels
 Arduino : arduino.cc
 ESP32 : espressif.com
 IEEE Robotics : ieee-ras.org



Code open-source
 Explorez des projets sur GitHub et inspirez-vous du travail des autres.



Vidéos et tutoriels
 Continuez à apprendre avec des tutoriels vidéo de qualité.



Communautés
 Rejoignez des forums, groupes et serveurs Discord pour échanger.

Scannez ce QR code pour accéder à la liste complète des ressources et liens utiles.



PARTAGEZ VOTRE SAVOIR
 Aidez la communauté en partageant vos projets, vos solutions et vos connaissances.



INNOVEZ CHAQUE JOUR
 Chaque petite amélioration peut mener à une grande innovation.



CONSTRUISEZ LE FUTUR
 Le futur de la robotique est entre vos mains ! Allez-y, construisez-le !

À très bientôt dans vos projets !



GLOSSAIRE



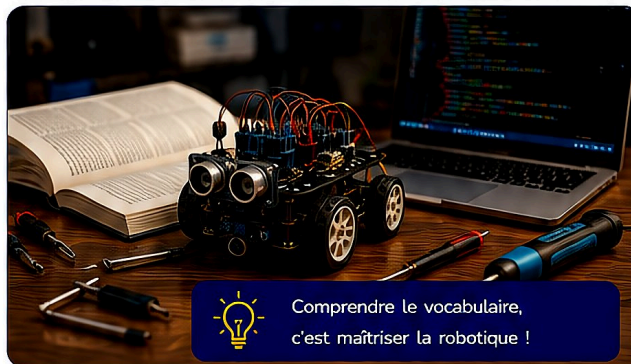
Ce glossaire regroupe les termes essentiels utilisés dans ce livre.

Gardez-le sous la main pour mieux comprendre la robotique et ses technologies.

123

GLOSSAIRE

Voici les principaux termes techniques rencontrés dans la robotique et leur définition simplifiée pour mieux les comprendre et les utiliser.



Comprendre le vocabulaire, c'est maîtriser la robotique !



Robot : Machine programmable capable d'effectuer des tâches automatiquement ou semi-automatiquement.



Breadboard : Plaque d'essai sans soudure permettant de prototyper des circuits électroniques.



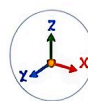
Automatisation : Utilisation de systèmes et de technologies pour effectuer des tâches sans intervention humaine.



Arduino : Carte électronique ouverte utilisée pour créer des projets interactifs et contrôler des systèmes.



Résistance : Composant électronique qui limite le courant dans un circuit. Unité : ohm (Ω).



Degré de liberté (DOF) : Nombre de mouvements indépendants qu'un robot peut effectuer.



ESP32 : Microcontrôleur puissant avec Wi-Fi et Bluetooth intégrés, idéal pour l'IoT et les projets avancés.



Condensateur : Composant qui stocke l'énergie électrique temporairement. Unité : farad (F).



Cinématique : Étude du mouvement des robots sans considérer les forces qui le provoquent.



Capteur : Dispositif qui mesure une grandeur physique (distance, lumière, température, etc.) et la convertit en signal électrique.



Transistor : Composant électronique utilisé pour amplifier ou commuter des signaux électriques.



Contrôle : Processus qui permet d'obtenir le comportement souhaité d'un système ou d'un robot.



Actionneur : Composant qui convertit l'énergie (électrique, pneumatique, hydraulique) en mouvement ou en action.



Moteur : Dispositif qui convertit l'énergie électrique en mouvement rotatif ou linéaire.



Sécurité : Ensemble des mesures prises pour protéger les personnes, les machines et les données.



Code : Ensemble d'instructions écrites dans un langage de programmation pour contrôler un système.



PWM : Modulation de largeur d'impulsion utilisée pour contrôler la vitesse ou la puissance des actionneurs.



IoT : Internet des objets, réseau de dispositifs connectés qui échangent des données via Internet.



Alimentation : Source d'énergie électrique qui fournit la tension et le courant nécessaires au fonctionnement d'un système.



API : Interface de programmation qui permet à différents logiciels de communiquer entre eux.



3D (Impression 3D) : Technologie permettant de fabriquer des objets physiques couche par couche à partir d'un modèle numérique.



Mécanique : Ensemble des pièces et structures qui permettent le mouvement et la transmission des forces.



Batterie : Dispositif qui stocke l'énergie électrique pour alimenter les systèmes lorsque nécessaire.



Débogage : Processus qui consiste à détecter et corriger les erreurs dans un programme ou un système.

LANGAGES DE PROGRAMMATION COURAMMENT UTILISÉS



Arduino (C/C++)

Langage principal pour programmer les cartes Arduino. Simple et efficace pour la robotique.



Python

Langage polyvalent, facile à apprendre, idéal pour l'IA, les données et l'automatisation.



C / C++

Langages puissants et rapides, utilisés pour les systèmes embarqués et la robotique.



MicroPython

Version allégée de Python pour microcontrôleurs comme ESP32 et micro:bit.



CONSEILS POUR BIEN UTILISER CE GLOSSAIRE

- ✓ Relisez régulièrement les termes pour les mémoriser.
- ✓ Revenez-y dès que vous rencontrez un nouveau concept.

- ✓ Utilisez ces mots dans vos projets et vos discussions.
- ✓ La pratique et la curiosité sont vos meilleurs alliés !



RÉFÉRENCE RAPIDE

Ce glossaire est votre guide rapide pour comprendre les concepts clés de la robotique.



PRATIQUEZ

Appliquez ces termes dans vos projets pour renforcer vos connaissances.



REJOIGNEZ LA COMMUNAUTÉ

Échangez avec d'autres passionnés, posez vos questions et partagez vos réalisations !

Scannez ce QR code pour accéder à des ressources, exemples de code et tutoriels.





**AU REVOIR
ET À BIENTÔT !**



Nous arrivons
au terme de ce
voyage passionnant
dans l'univers de
la robotique.
Merci de votre
confiance !

Continuez à apprendre,
à créer et surtout,
à ne jamais cesser
de rêver.

**Le futur appartient
à ceux qui
construisent aujourd'hui !**

124

AU REVOIR ET À BIENTÔT !

Félicitations ! Vous avez terminé ce livre sur la robotique moderne. Vous avez maintenant toutes les bases nécessaires pour comprendre, concevoir et réaliser vos propres robots en utilisant l'électronique, la programmation et la créativité.

Mais souvenez-vous : ce n'est que le début de votre aventure !

“ La robotique est un chemin sans fin où chaque projet vous rapproche un peu plus de l'innovation.

Continuez à explorer, à expérimenter et à construire le monde de demain !



**Chaque idée peut devenir
une invention qui change le monde.**

RESSOURCES POUR ALLER PLUS LOIN



Livres recommandés

- Robotique pour tous – P. Corke
- Programming Robots with ROS
- Arduino Cookbook



Chaînes YouTube

- GreatScott! • DroneBot Workshop
- Andreas Spiess • The Construct



Sites web utiles

- arduino.cc – Documentation officielle
- learn.adafruit.com – Tutoriels
- robotics.stackexchange.com – Forum



Logiciels et outils

- Arduino IDE • PlatformIO • Visual Studio Code
- Tinkercad • Fusion 360 • ROS



Communautés

- Discord (robotique, Arduino, ESP32)
- Reddit r/robotics • Hackaday.io

IDÉES DE PROJETS À RÉALISER



Robot suiveur de ligne

Utilise des capteurs IR et un microcontrôleur.



Bras robotique 3D

Contrôlé par servomoteurs et Arduino.



Drone quadrirotor

Apprenez l'électronique de puissance et le contrôle de vol.



Maison intelligente

Contrôlez l'éclairage, la température et la sécurité.



Rover d'exploration

Robot tout-terrain avec caméra et télémétrie.

VOTRE PARCOURS DE APPRENTISSAGE

- Découverte**
Vous avez découvert le monde fascinant de la robotique.
- Apprentissage**
Vous avez acquis des connaissances en électronique, programmation et mécanique.
- Pratique**
Vous avez réalisé des projets concrets et développé vos compétences.
- Maîtrise**
Vous êtes prêt à relever de nouveaux défis et à innover.
- Inspiration**
Partagez vos idées et inspirez les autres autour de vous.

VOS PREMIERS PAS VERS L'AVENIR



Fixez-vous
des objectifs
clairs.



Continuez à
expérimenter et
à tester.



Participez à des
communautés
et événements.



Travaillez sur
des projets qui
vous passionnent.



Ne craignez pas
l'échec, il fait partie
de l'apprentissage.



Rêvez grand
et construisez
le futur !



La meilleure façon de prédire le futur est de l'inventer.

– Alan Kay



MERCI !

Merci d'avoir choisi ce livre pour vous accompagner dans votre apprentissage de la robotique. Nous espérons qu'il vous a été utile, inspirant et motivant.

À très bientôt pour d'autres projets incroyables !

PARTAGEZ VOTRE EXPÉRIENCE



Vous avez aimé ce livre ? Laissez un avis, partagez-le avec vos amis et rejoignez notre communauté pour continuer l'aventure ensemble !

TÉLÉCHARGEZ LES RESSOURCES



Scannez le QR code pour accéder à la liste complète des codes sources, schémas, fiches techniques et tutoriels supplémentaires.



SCANNEZ-MOI !



À VOUS DE JOUER !

Le monde a besoin de vos idées, de votre créativité et de vos robots.



APPRENEZ. CRÉEZ. PARTAGEZ.

CONSTRUISONS ENSEMBLE LE MONDE DE DEMAIN !



AU REVOIR
ET À
BIENTÔT !



Ce n'est pas la fin,
mais le début d'une
belle aventure.

Continuez à apprendre,
à créer, à tester
et à innover.

La robotique
a besoin de vous !

Merci d'avoir choisi
ce livre pour vous
accompagner sur
ce chemin passionnant.

125

MERCI ET BONNE CONTINUATION !

Nous espérons que ce livre vous a apporté des connaissances utiles et vous a donné l'envie d'aller encore plus loin dans le fascinant univers de la robotique.

Le progrès en robotique ne s'arrête jamais, et chaque jour apporte de nouvelles idées, de nouveaux défis et de nouvelles solutions.

“ L'avenir appartient à ceux
qui croient en la beauté de leurs rêves.

– Eleanor Roosevelt



Continuez à rêver, à construire
et à changer le monde avec vos robots !

À TRÈS BIENTÔT DANS VOS PROJETS !

CE QUE VOUS POUVEZ FAIRE ENSUITE

- ✓ Relisez les chapitres et révisez les concepts importants.
- ✓ Reprenez les projets et personnalisez-les pour aller plus loin.
- ✓ Explorez de nouvelles technologies (IA, vision, IoT, ROS, impression 3D...).
- ✓ Partagez vos créations et échangez avec la communauté.
- ✓ Ne cessez jamais d'être curieux !



Chaque petite étape
vous rapproche de
grandes réalisations !

RAPPEL DES COMPÉTENCES ACQUISES



Comprendre les principes de la robotique, de l'électronique et du contrôle.



Programmer des cartes Arduino et ESP32 et communiquer avec des capteurs.



Utiliser des actionneurs et concevoir des systèmes automatisés.



Concevoir, tester et améliorer des projets concrets.



Intégrer la sécurité et la fiabilité dans vos créations.

VOTRE BOÎTE À OUTILS À GARDER



Arduino IDE



ESP32 / PlatformIO



Capteurs et actionneurs



Bibliothèques & librairies



Outils : Multimètre, Oscilloscope, Imprimante 3D



Sauvegardes et documents de référence

IDÉES DE DOMAINES À EXPLORER



Industrie
Automatisation
et contrôle



Médecine
Aide chirurgicale
et rééducation



Agriculture
Robots autonomes
et surveillance



Exploration
Robots spatiaux
et sous-marins



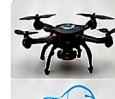
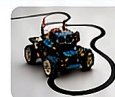
Services
Robots domestiques
et assistance



Éducation
Recherche et
innovation

EXEMPLES DE PROJETS À POURSUIVRE

- ✓ Robot suiveur de ligne avancé
- ✓ Bras robotique avec retour de force
- ✓ Voiture autonome évitant les obstacles
- ✓ Drone avec stabilisation et caméra
- ✓ Station météo connectée
- ✓ Maison intelligente IoT
- ✓ Robot télécommandé avec IA embarquée



N'oubliez pas : la persévérance, la pratique et la passion sont les clés du succès en robotique.

Continuez à apprendre, à innover et à inspirer !



MERCI !

Merci d'avoir choisi ce livre.
Nous vous souhaitons beaucoup de succès
dans vos projets et vos études.
**Le futur de la robotique
est entre vos mains !**



RESSOURCES UTILES



Sites et documentation
arduino.cc
esp32.com
docs.ros.org



Vidéos et tutoriels
Chaines YouTube
GreatScott!, DroneBot,
The Construct, etc.



Code open-source
Explorez, apprenez
et partagez sur
GitHub.



Communautés
Discord, Reddit, forums
Arduino, ESP32 et
robotique.



Livres recommandés
Approfondissez vos
connaissances avec les
ouvrages spécialisés.

Scannez ce QR code
pour accéder à la liste
complète des ressources
et bonus en ligne !



VOTRE AVENTURE NE FAIT QUE COMMENCER !

Rêvez grand, codez mieux, construisez l'avenir.



APPRENEZ. CRÉEZ. PARTAGEZ.

CONSTRUISONS ENSEMBLE LE MONDE DE DEMAIN !



À PROPOS DE DEX ATOMES



Dex Atomes est une entreprise passionnée par l'innovation, la technologie et l'éducation.

“Notre mission est simple : rendre la robotique et les technologies avancées accessibles à tous, pour créer un avenir meilleur.”



126

À PROPOS DE DEX ATOMES

INNOVER. ÉDUIQUER. INSPIRER.

Dex Atomes est une entreprise dédiée à la création de ressources éducatives et de solutions technologiques innovantes dans le domaine de l'ingénierie, de la robotique et de l'automatisation.

Nous croyons que la connaissance est le moteur du progrès et que chaque personne, avec les bons outils et les bonnes compétences, peut transformer des idées en réalités capables de changer le monde.

Ce livre est le reflet de cette vision.



INNOVATION

Nous explorons, concevons et développons des solutions intelligentes pour relever les défis technologiques d'aujourd'hui et de demain.



ÉDUCATION

Nous créons des contenus pédagogiques clairs, pratiques et accessibles pour aider les étudiants, les professionnels et les passionnés à apprendre et à progresser.



ROBOTIQUE & IA

Nous nous concentrons sur la robotique, l'intelligence artificielle et l'automatisation pour construire un avenir où les machines et les humains collaborent intelligemment.



COMMUNAUTÉ

Nous croyons au partage des connaissances, à l'entraide et à la création d'une communauté mondiale de bâtisseurs et d'innovateurs.



DEX ATOMES
TECHNOLOGIE • ÉDUCATION • INNOVATION

DEX ATOMES EN QUELQUES MOTS

Nous concevons des outils, des formations et des projets qui permettent à chacun d'apprendre en faisant, de créer en expérimentant et d'innover en repoussant les limites de l'imagination.

Apprendre aujourd'hui, construire demain.



VISION GLOBALE

Des ressources pour tous, partout dans le monde.



APPRENANTS

Des milliers d'étudiants et de passionnés nous font confiance.



PROJETS

Des projets concrets pour mettre en pratique vos idées.

NOS VALEURS



INTÉGRITÉ

Nous agissons avec honnêteté, transparence et respect dans tout ce que nous faisons.



EXCELLENCE

Nous visons la qualité dans nos contenus, nos produits et nos services.



PASSION

Nous sommes animés par la passion de la technologie et de l'enseignement.



CRÉATIVITÉ

Nous encourageons l'innovation, l'expérimentation et la pensée créative.



COLLABORATION

Nous avançons ensemble vers un objectif commun : un monde meilleur.

NOTRE AVENIR

Dex Atomes continue d'investir dans l'éducation technologique et la recherche pour offrir des solutions toujours plus puissantes et accessibles.

Rejoignez-nous dans cette aventure et construisons ensemble le futur de la robotique !



Le meilleur moyen de prédire le futur, c'est de le créer.

– Peter Drucker –



Apprendre



Construire



Innover



Inspirer



DÉCOUVREZ
DEXUS LAB



DEXUS LAB
Learn. Build. Innovate.

Dexus Lab est la plateforme tout-en-un pour apprendre, pratiquer et maîtriser les compétences techniques les plus recherchées dans les domaines de l'ingénierie, de la robotique, de l'IA et des technologies émergentes.

Apprenez à votre rythme. Construisez vos projets. Obtenez votre certification.



127

VOTRE PLATEFORME D'APPRENTISSAGE EN INGÉNIERIE ET TECHNOLOGIE

Dexus Lab vous accompagne du débutant à l'expert grâce à des cours structurés, des projets pratiques, des défis et des certifications reconnues.



COURS STRUCTURÉS

Des leçons claires avec vidéos, ressources et exemples pratiques.



CERTIFICATS

Obtenez des certificats pour valoriser vos compétences.



CODE & SIMULATION

Éditeur de code intégré, simulations et exercices interactifs.



SUIVI DE PROGRESSION

Suivez vos résultats, statistiques et améliorations.



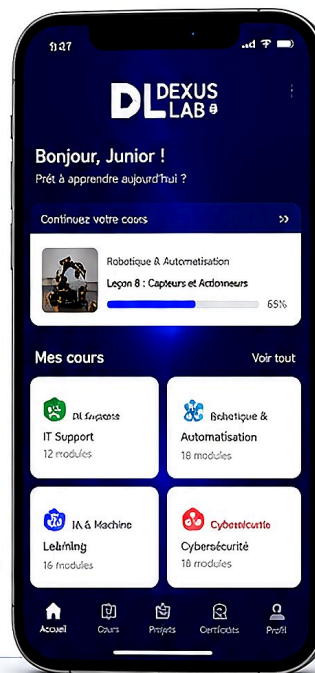
DÉFIS & PROJETS

Mettez en pratique vos compétences avec des projets réels et défis.

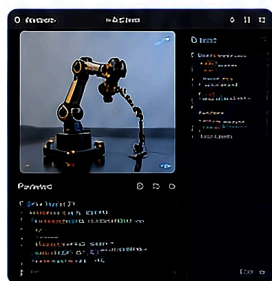


ACCÈS PARTOUT

Apprenez sur mobile, tablette et ordinateur, en ligne ou hors ligne.



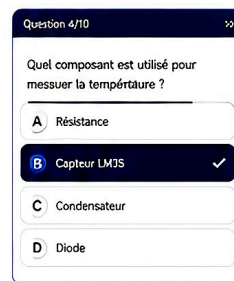
UNE EXPÉRIENCE MODERNE ET INTUITIVE



Leçons vidéo interactives



Éditeur de code intégré



Quiz et évaluations



Certificats reconnus

NOS COURS PHARES

1. IT SUPPORT



Maintenance, réseaux, dépannage et support informatique.

2. ROBOTIQUE & AUTOMATISATION



Conception, programmation et contrôle de systèmes robotiques.

3. IA & MACHINE LEARNING



Apprenez l'IA, le machine learning et la data science par la pratique.

4. CYBERSÉCURITÉ



Protégez les systèmes, analysez les menaces et apprenez l'éthique hacking.

5. DRONES & UAV



Conception, pilotage et applications professionnelles des drones.

6. PYTHON POUR INGÉNIEURS



Apprenez Python pour l'automatisation, l'analyse de données et plus.

7. EMBEDDED C & IOT



Programmation C pour systèmes embarqués et projets IoT connectés.

8. PROJETS & INNOVATION



Développez des projets réels et transformez vos idées en solutions concrètes.

POURQUOI CHOISIR DEXUS LAB ?

- ✓ Contenu à jour et orienté vers l'industrie
- ✓ Apprentissage pratique et par projets
- ✓ Communauté active d'apprenants
- ✓ Accès illimité à tous les cours Premium
- ✓ Support et mentorat dédiés

REJOIGNEZ DES MILLIERS D'APPRENANTS DÉJÀ ACTIFS !



10 000+ Apprenants dans le monde

PRÊT À COMMENCER ?

Téléchargez Dexus Lab et commencez votre aventure d'apprentissage dès aujourd'hui !



Disponible sur
App Store



DISPONIBLE SUR
Google Play



APPRENDRE

Suivez des cours clairs et complets.



CONSTRUIRE

Réalisez des projets et expérimentez.



RÉUSSIR

Obtenez des certificats et valorisez vos compétences.

SCANNEZ-MOI !



www.dexuslab.com



NOS FORMATIONS ET CERTIFICATIONS



Dexus Lab vous offre des parcours complets, des compétences reconnues et des certificats valorisables pour booster votre avenir professionnel.

“Investir dans votre formation aujourd’hui, c’est construire votre succès de demain.”

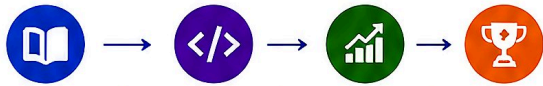
128

DES PARCOURS, DES COMPÉTENCES, DES CERTIFICATIONS RECONNUES

Nos formations sont conçues par des experts pour vous offrir une expérience d'apprentissage complète, pratique et certifiante.



PARCOURS D'APPRENTISSAGE STRUCTURÉS



1. APPRENDRE

Suivez des cours structurés avec des leçons claires, des exemples et des ressources pratiques.

2. PRATIQUER

Réalisez des exercices, projets et simulations pour appliquer vos connaissances dans des situations réelles.

3. ÉVALUER

Testez vos compétences avec des quiz et des défis. Atteignez au moins 70 % pour valider chaque étape.

4. CERTIFIER

Obtenez votre certificat et valorisez vos compétences auprès des employeurs et du monde entier.

NOS FORMATIONS DISPONIBLES



IT SUPPORT

Dépannage, réseaux et maintenance



DESIGN & FLY YOUR DRONE

Conception, pilotage et missions



ROBOTIQUE & AUTOMATISATION

Conception et contrôle de systèmes robotisés



CYBERSÉCURITÉ EN PRATIQUE

Protéger les systèmes et les données



IA & MACHINE LEARNING

Algorithmes, modèles et IA appliquée



C PROGRAMMING EMBEDDED SYSTEMS

C pour systèmes embarqués et IoT



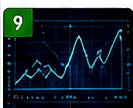
PYTHON POUR INGÉNIEURS

Automatisation, analyse et scripting



STRUCTURAL ANALYSIS

Analyse et conception des structures



CONTROL SYSTEMS ENGINEERING

Modélisation, stabilité et régulation



ADVANCED CAD & SIMULATION

Conception et simulation avancées



POWER SYSTEMS & ENERGY

Énergie, réseaux et systèmes électriques



ENGINEERING PROJECT MANAGEMENT

Planification, gestion et suivi de projets

CERTIFICATS DE RÉUSSITE

Chaque formation complétée avec succès vous permet d'obtenir un certificat numérique personnalisé.



Vérifiables en ligne

Certificats sécurisés et authentifiables.



Téléchargeables

Obtenez vos certificats en PDF.



Partageables

Partagez sur LinkedIn, CV et réseaux sociaux.

LES AVANTAGES PREMIUM



Accès illimité à tous les cours et ressources.



Téléchargement de ressources et exercices.



Support prioritaire et mentorat avancé.



Nouveaux cours et contenus exclusifs.

MISES À JOUR RÉGULIÈRES

De nouveaux contenus, projets et technologies sont ajoutés régulièrement pour vous garder à la pointe du progrès.



À VENIR : PROCHAINES FORMATIONS

PROCHAINEMENT



ROBOTIQUE MOBILE AVANCÉE

Navigation, SLAM et perception.

PROCHAINEMENT



IOT & SYSTÈMES CONNECTÉS

Capteurs, réseaux et applications IoT.

PROCHAINEMENT



IMPRESSION 3D & SIMULATION

Modélisation, impression et prototypage.

PROCHAINEMENT



RÉALITÉ VIRTUELLE & SIMULATION

Simulation immersive et applications.

PRÊT À PASSER AU NIVEAU SUPÉRIEUR ?

- ✓ Choisissez votre parcours
- ✓ Apprenez à votre rythme
- ✓ Réalisez des projets concrets
- ✓ Obtenez des certificats reconnus
- ✓ Construisez votre avenir



Rejoignez Dexus Lab et transformez votre passion en expertise !

www.dexuslab.com



L'éducation est l'arme la plus puissante que vous pouvez utiliser pour changer le monde. – Nelson Mandela





REJOIGNEZ
LA COMMUNAUTÉ
DEXUS LAB !



Apprendre ensemble,
s'entraider, partager
ses projets et grandir
chaque jour : c'est
la force de notre
communauté.

"Seul on va plus vite,
ensemble on va
plus loin."

– Proverbe africain

129

REJOIGNEZ NOTRE COMMUNAUTÉ ET RESTEZ CONNECTÉ !

La robotique est une aventure collective. Rejoignez des milliers d'apprenants, de passionnés et de professionnels qui partagent la même curiosité et la même envie d'innover.

Échangez. Apprenez. Inspirez. Construisons l'avenir ensemble !



NOS ESPACES COMMUNAUTAIRES



DISCORD

Discutez, posez vos questions, partagez vos projets et obtenez de l'aide en temps réel.

discord.gg/dexuslab



FORUM DEXUS LAB

Participez aux discussions, lisez des tutoriels avancés et trouvez des solutions à vos problèmes.

forum.dexuslab.com



YOUTUBE

Tutoriels, projets, lives et présentations de robots. Apprenez en vidéo avec des contenus exclusifs.

youtube.com/@dexuslab



GITHUB

Accédez à nos codes sources, bibliothèques et exemples de projets open-source.

github.com/dexus-lab



NEWSLETTER

Recevez chaque mois des conseils, articles et ressources exclusives directement par email.

dexuslab.com/newsletter

SUIVEZ-NOUS SUR LES RÉSEAUX SOCIAUX



facebook.com/dexuslab
Actualités, événements et nouveautés



instagram.com/dexuslab
Projets, photos et coulisses



tiktok.com/@dexuslab
Vidéos courtes, astuces et démonstrations



linkedin.com/company/dexus-lab
Réseau professionnel et opportunités

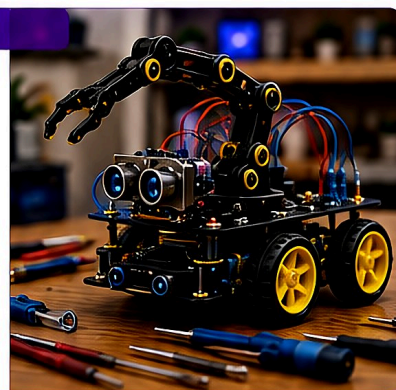


x.com/dexuslab
Actualités tech et annonces rapides

PARTAGEZ VOS PROJETS !

Montrez ce que vous créez et inspirez les autres membres de la communauté !

- ✓ Publiez vos projets
- ✓ Recevez des retours
- ✓ Collaborez avec d'autres passionnés
- ✓ Mettez en avant vos idées et innovations



BESOIN D'AIDE ?



DOCUMENTATION

Guides, FAQ et tutoriels détaillés sur notre site.

dexuslab.com/docs



SUPPORT

Contactez notre équipe pour toute question technique ou compte.

support@dexuslab.com



CHAT EN DIRECT

Discutez avec nous directement depuis l'application ou le site.

dexuslab.com/chat



SUGGESTIONS

Une idée ? Une amélioration ? Votre avis compte pour nous faire progresser !

dexuslab.com/ideas

RESTEZ INFORMÉ



Activez les notifications pour ne rien manquer :

- ✓ Nouveaux cours
- ✓ Mises à jour
- ✓ Événements en direct
- ✓ Offres exclusives



La communauté est le cœur de l'apprentissage : elle transforme la connaissance en expérience partagée.

– Henry Ford



Merci de faire partie de la communauté Dexus Lab !
Continuons d'apprendre, d'innover et de construire le monde de demain.

À très bientôt dans nos cours, projets et discussions !





À PROPOS DE L'AUTEUR

Jean-Claude Junior Desulme

Passionné par la technologie, l'ingénierie et l'innovation, Jean-Claude Junior Desulme a créé ce livre pour partager ses connaissances et inspirer la prochaine génération d'ingénieurs, de créateurs et de bâtisseurs.

Jean-Claude Junior Desulme

Fondateur de Dex Atomes

Créateur de Dexus Lab

DX
DEX ATOMES

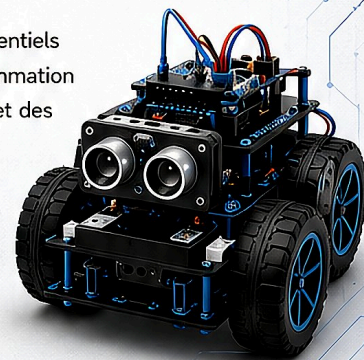
ROBOTIQUE MODERNE

GUIDE COMPLET POUR DÉBUTANTS

ARDUINO • ESP32 • IA • DRONES • CAPTEURS • PROJETS PRATIQUES

Ce livre vous a guidé à travers les concepts essentiels de la robotique, de l'électronique, de la programmation Arduino et ESP32, de l'intelligence artificielle et des projets pratiques.

Continuez à apprendre, à expérimenter et à construire. L'avenir de la technologie se crée aujourd'hui, et vous en faites partie !



DANS CE LIVRE, VOUS AVEZ APPRIS À :



COMPRENDRE

les bases de la robotique et de l'électronique



PROGRAMMER

Arduino, ESP32 et utiliser des capteurs



CONCEVOIR

et construire des projets pratiques



EXPLORER

l'IA, les drones et les technologies émergentes



DÉVELOPPER

vos compétences et réaliser vos idées

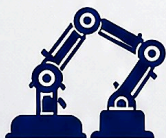
“

L'innovation distingue un leader d'un suiveur.

– Steve Jobs

CONTINUEZ VOTRE APPRENTISSAGE

- ✓ Pratiquez régulièrement
- ✓ Rejoignez une communauté
- ✓ Partagez vos projets
- ✓ Ne cessez jamais d'innover



DÉCOUVREZ DEXUS LAB

La plateforme d'apprentissage en ingénierie et technologie créée par Dex Atomes.



COURS
STRUCTURÉS



PROJETS
PRATIQUES



CERTIFICATS
RECONNUS



COMMUNAUTÉ
ACTIVE

Téléchargez Dexus Lab sur



SUJETS COUVERTS

- Robotique & Mécatronique
- Arduino & ESP32
- Capteurs & Actionneurs
- IA & Machine Learning
- Drones & Systèmes embarqués
- Projets pratiques étape par étape



DE LA THÉORIE À LA PRATIQUE :
APPRENEZ, COMPRENEZ ET CRÉEZ VOS PROPRES ROBOTS !

Apprenez aujourd'hui, construisez demain.

ISBN 978-2-4877-0256-0



Tous droits réservés.