

# ARCHITECTURE ELECTRONIQUE DE ROBOT

AXELLE MAMPAH  
RONG ZHOU

## ➤ INTRODUCTION

## ➤ DEVELOPPEMENT

### I- Les constituants du robot chenille

#### A- Source d'énergie

#### B- Contrôle

#### C- Unités motrices

#### D- Accessoires

### II- Programmation et microprocesseur

### III- Programme avec les conditions

### IV- Utilisation

## ➤ CONCLUSION

## ➤ INTRODUCTION

Aujourd'hui, nous allons vous présenter le robot chenille. Il s'agit d'un robot mobile qui se déplace grâce à des chenilles, comme celles des chars ou des engins de chantier. Ce type de robot est particulièrement adapté aux terrains difficiles, car il peut avancer sur le sable, la boue, les gravats ou les pentes. Le robot chenille est utilisé dans plusieurs domaines, notamment l'exploration, le sauvetage, l'industrie et même le domaine militaire. Dans cet exposé, nous allons vous expliquer de quoi est composé ce robot, de sa programmation et son utilisation.

## I- Les constituants du robot chenille

Le robot chenille est constitué de :

### A- Source d'énergie

1. Boîtier pour piles AAA (4)
  - 2 orientées vers le bas
  - 2 orientées vers le haut

Interrupteur principal (1)

### B- Contrôle

#### - **Entrées / Sorties (I/O)**

2. Port USB (1): vers PC
3. Port série (1) : vers la carte mère

#### - **Carte mère**

1. Microprocesseur: ATMEGA16A-AU 1326 (1)
2. Bouton Start/Stop (1)

#### - **Capteurs**

1. Capteur de roulis (1)

### C- Unités motrices

- 1- Actionneurs (8)
  - Mouvement vertical (4)
  - Mouvement latéral (4)
- 2- Patins de pieds ( $5 \times 2 = 10$ )

### D- Accessoires

### **Éclairage**

- 2. Lumières d'antennes (3)
  - 1. Tête (2)
  - 2. Queue (1)
- 3. Feu de position (1)
  - 1. Côté gauche (1)

### **Câbles**

- 1. Divers câbles isolés reliant différents composants électroniques
  - génériques
  - $0,25 \text{ mm}^2$
  - La plupart des connexions sont soudées à la main

## II- Programmation et microprocesseur

### A – Origine du langage C

Le langage de programmation utilisée est le langage C. Développé au début des années 1970 dans les **Laboratoires Bell aux États-Unis** par **Dennis Ritchie et Kenneth Thompson**, le langage C a initialement été conçu pour programmer le système d'exploitation

Unix, dont une grande partie est encore aujourd'hui écrite en C. Cette technologie est directement issue du **langage B**, précédemment créé par Kenneth Thompson.

## B- Utilisation du langage C

Le langage C est un langage de programmation :

- **Compilé**, ce qui signifie que le code rédigé est traduit en langage machine par un compilateur, permettant au programme d'être exécuté très rapidement. À l'inverse, un langage interprété est lu et exécuté ligne par ligne ;
- **Impératif**, car il fonctionne étape par étape, en indiquant quelle instruction suivre et dans quel ordre ;
- **Structuré**, c'est-à-dire qu'il encourage le développeur à organiser son code de manière claire, en le divisant par blocs ou fonctions, afin de faciliter sa lecture, sa maintenance et sa réutilisation.

Le C est conçu pour **créer des programmes proches du matériel** tout en conservant un certain niveau d'**abstraction**. Il s'agit en effet d'un **langage de bas niveau**. Ce type de technologie est très proche du fonctionnement de la machine et permet de communiquer directement avec ses composants, comme la mémoire, le processeur ou les périphériques.

### III- Programme avec les conditions

Action	Moteur gauche	Moteur droit
Avancer	ON ON ON ON	ON ON ON ON
Tourner G	OFF OFF OFF ON	ON ON ON ON
Tourner D	ON ON ON ON	OFF OFF OFF ON

Notre robot est équipé avec les actionneurs. Les actionneurs sont contrôlé par le micro-processeur ATMEGA16A-AU 1326.

Un code exemple simple pour controller un actionneur :

```
#include <avr/io.h>
#include <util/delay.h>

// Convertit un angle (0-180°) en largeur d'impulsion (OCR1A)
uint16_t angle_to_ocr(int angle) {
    // 1 ms = 1000 µs → 1000 * 2 = 2000 cycles (à 8 MHz / prescaler 8)
    // 2 ms = 2000 µs → 4000 cycles
    return 2000 + (angle * 2000) / 180;
}

int main(void) {
    // PB1 = OC1A → sortie PWM
    DDRB |= (1 << PB1);

    // Timer1 en mode Fast PWM, TOP = ICR1
    TCCR1A = (1 << COM1A1) | (1 << WGM11);
    TCCR1B = (1 << WGM13) | (1 << WGM12) | (1 << CS11); // prescaler 8

    // Période PWM = 20 ms → ICR1 = 20000 µs * (F_CPU / prescaler)
    // F_CPU = 8 MHz → 1 µs = 8 cycles → 20 ms = 20000 µs → 20000 * 1 = 20000
    ICR1 = 20000;

    while (1) {
        // Servo à 0°
        OCR1A = angle_to_ocr(0);
        _delay_ms(1000);

        // Servo à 90°
        OCR1A = angle_to_ocr(90);
        _delay_ms(1000);

        // Servo à 180°
        OCR1A = angle_to_ocr(180);
        _delay_ms(1000);
    }
}
```

Un exemple qui contrôle tous les 8 actionneurs ensemble :

```

#include <avr/io.h>
#include <util/delay.h>

#define SERVO_MIN 1000    // 1 ms
#define SERVO_MAX 2000    // 2 ms

// Convertit un angle (0-180°) en durée d'impulsion (µs)
uint16_t angle_to_us(uint8_t angle) {
    return SERVO_MIN + (angle * (SERVO_MAX - SERVO_MIN)) / 180;
}

int main(void) {
    // 8 servos sur PB0..PB7
    DDRB = 0xFF; // tout le port B en sortie

    // Angles initiaux des 8 servos
    uint8_t angles[8] = {90, 90, 90, 90, 90, 90, 90, 90};

    while (1) {
        uint16_t total_time = 0;

        for (uint8_t i = 0; i < 8; i++) {
            uint16_t pulse = angle_to_us(angles[i]);

            // Mettre à 1 la broche du servo i
            PORTB = (1 << i);
            _delay_us(pulse);

            // Remettre à 0
            PORTB = 0x00;

            total_time += pulse;
        }

        // Compléter le cycle de 20 ms
        if (total_time < 20000)
            _delay_us(20000 - total_time);

        // Exemple : faire bouger le servo 0
        angles[0] = (angles[0] + 5) % 180;
    }
}

```

## IV-Utilisation du robot chenille

Les robots chenillés sont conçus pour se déplacer efficacement sur des terrains difficiles grâce à leurs chenilles, qui leur offrent une grande stabilité, une bonne adhérence et la capacité de franchir des obstacles comme les pentes, les escaliers ou les débris. Ils jouent un rôle



essentiel dans des domaines où les conditions sont dangereuses ou complexes, notamment dans le déminage, les opérations de secours après des catastrophes naturelles, l'inspection de sites industriels sensibles et certaines missions militaires. En réduisant les risques pour l'être humain, ces robots permettent d'explorer, d'analyser et d'intervenir dans des zones difficiles d'accès tout en transportant des capteurs, des caméras ou des outils spécialisés.

## ➤ Conclusion

En conclusion, les robots chenillés sont des outils essentiels pour intervenir dans des environnements difficiles et dangereux. Grâce à leur mobilité et leur stabilité, ils permettent d'accomplir des missions complexes tout en limitant les risques pour l'être humain, ce qui en fait une technologie de plus en plus importante dans de nombreux domaines.

## Specification de robot

- DOF (degrés de liberté): 8
- programmable
- plusieurs portes d'entrée pour les capteurs supplémentaires

Élément	Valeur
Alimentation	5,2 V à 6 V (4 piles AAA, non incluses)
Processeur	ATMEGA16
Servomoteurs	8 mini-servomoteurs
Consommation	~100 mA au repos, jusqu'à 3 A avec tous les servos en action
Hauteur	90 mm
Longueur	500 mm
Largeur	85 mm