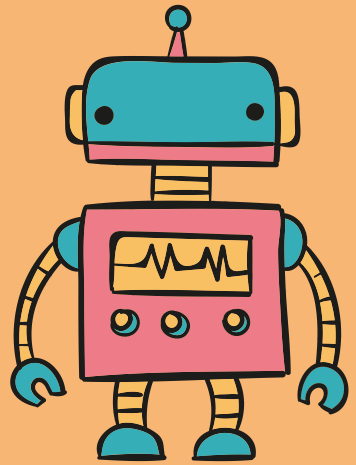
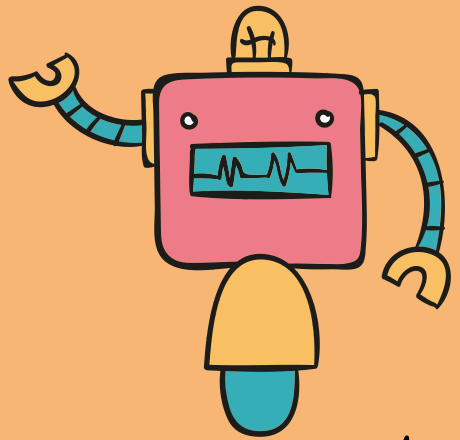
PROJET ROBOT 2

Soutenance

Théo GACHET Cyril HÉRAIL



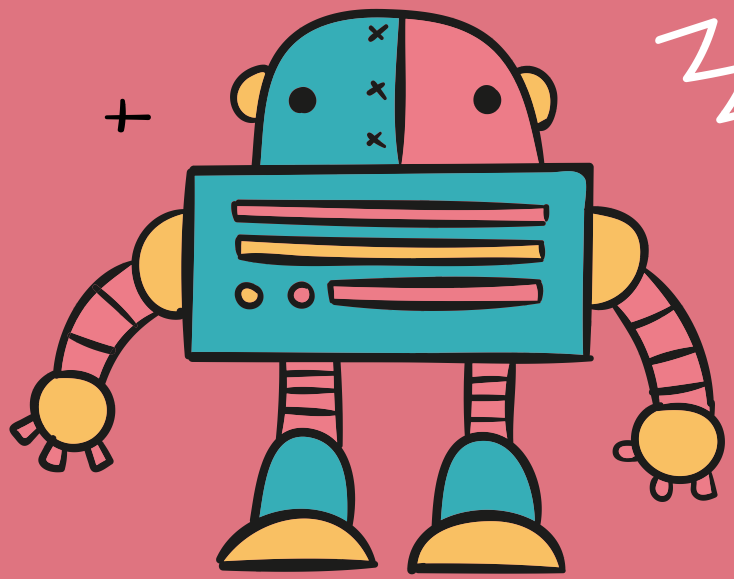


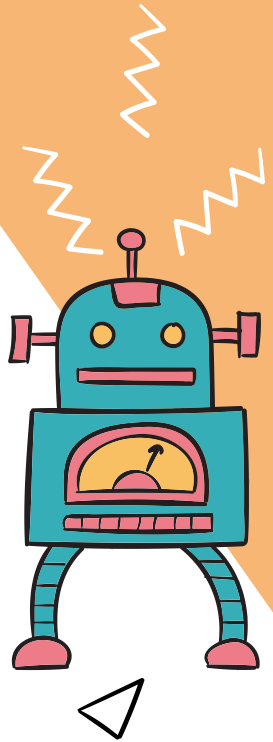
Introduction

Cours de Systèmes à Microcontrôleurs

Projet Robot 1

Projet Robot 2





01

Presentation

Contexte et rapide présentation du contrat

02

Fonctions du robot

Fonctionnalités de notre code et différents modules

03

Resultats et ameliorations

Résultats des simulations, améliorations éventuelles, difficultés

04

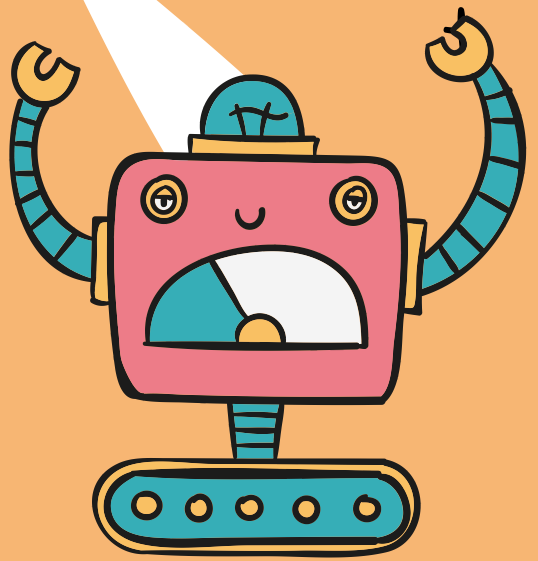
Conclusion

Ce que nous avons obtenu, retenu et apprécié

01.

Presentation

Contexte et rapide présentation du contrat



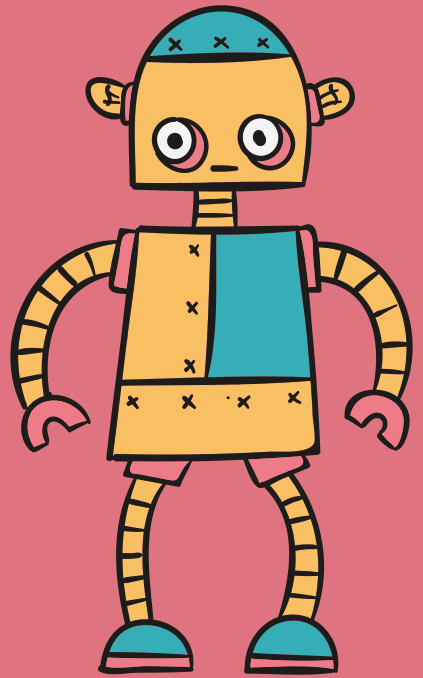
Contrat 7

Le robot doit se diriger vers une cible à l'avant du robot

Capteurs IR

Le robot démarre et avance en ligne droite s'il détecte un obstacle fixe à moins de 1,50 m et s'arrête si la distance est inférieure à 40 cm

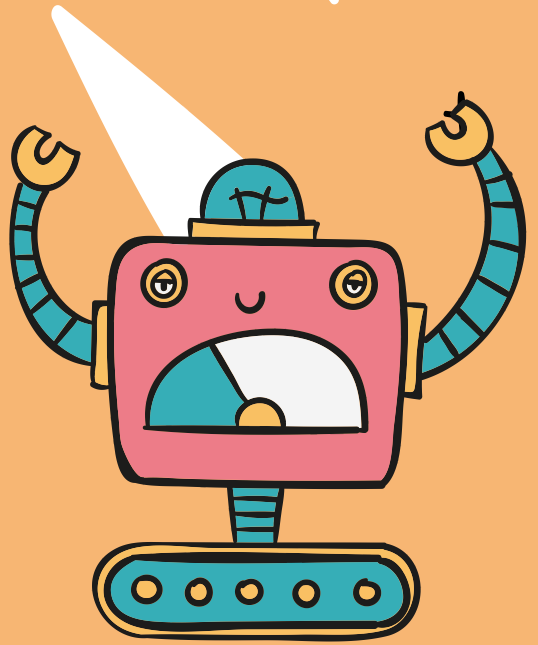
Acquisitions des signaux des capteurs : interruptions du débordement du Timer0 toutes les 20ms



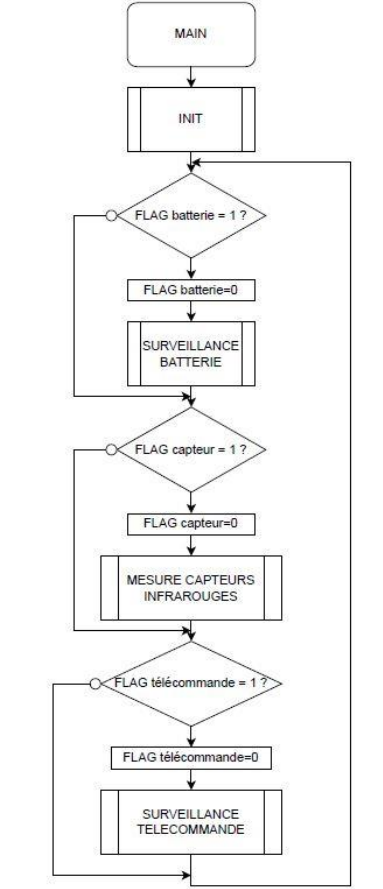
02.

Fonctions du robot

Algorigrammes et code



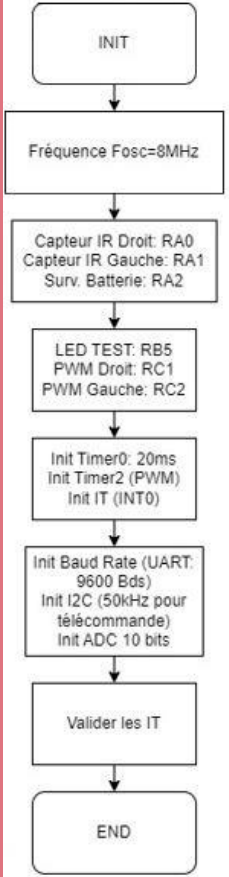
Algorigramme



Code

```c
void main(void)
{
    init_uart();
    init_frequence();
    init_entrees_sorties();
    init_moteurs();
    init_timer();
    init_I2C();
    init_ADC();
    valider_IT();

    while (1)
    {
        if (flag_batterie == 1)
        {
            flag_batterie = 0;
            surveillance_batt();
        }
        if (flag_capteurs_IR == 1)
        {
            flag_capteurs_IR = 0;
            surveillance_capt();
        }
        if (flag_telecommande == 1)
        {
            flag_telecommande = 0;
            surveillance_tele();
        }
    }
}
```

Algorigramme

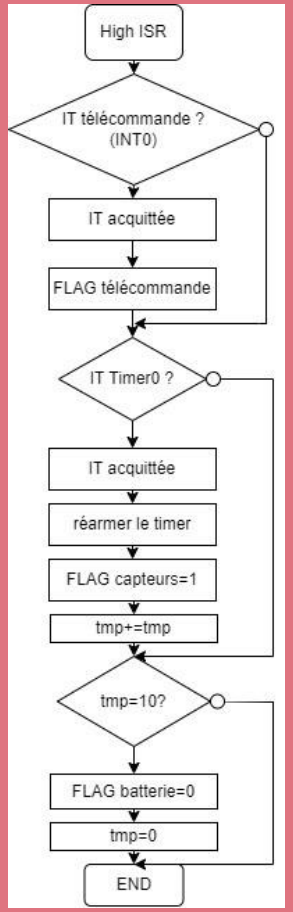


Code

```
void init_frequence(void) // fréquence d'horloge =
{
    OSCCONbits.IRCF0 = 1;
    OSCCONbits.IRCF1 = 1;
    OSCCONbits.IRCF2 = 1;
}

void init_entrees_sorties(void)
{
    // entrées
    TRISCbits.TRISC3 = 1; // SCL en entree (p176)
    TRISCbits.TRISC4 = 1; // SDA en entree (p176)
    TRISAbits.RA0 = 1;  // IR droit
    TRISAbits.RA1 = 1;  // IR gauche

    // sorties
    TRISCbits.RC1 = 0; // PWM Droit
    TRISCbits.RC2 = 0; // PWM Gauche
    TRISBbits.RB5 = 0; // LED Test
    TRISBbits.RB1 = 0; // IR_ON (alim)
    PORTBbits.RB1 = 0;
}

void init_moteurs(void)
{
    T2CONbits.T2CKPS1 = 0; // Prescaler = 4 (p135)
    T2CONbits.T2CKPS0 = 1;
    PR2 = 124;
    CCPR1L = 0; // le robot est initialement immob
    CCPR2L = 0;
    CCP1CONbits.DC1B0 = 0; // p149
    CCP1CONbits.DC1B1 = 0;
    CCP2CONbits.DC2B0 = 0;
    CCP2CONbits.DC2B1 = 0;

    CCP1CONbits.CCP1M3 = 1; // mode PWM (p146)
    CCP1CONbits.CCP1M2 = 1;
    CCP2CONbits.CCP2M3 = 1;
    CCP2CONbits.CCP2M2 = 1;

    T2CONbits.TMR2ON = 1; // Timer2
    T2CONbits.T2OUTPS = 9;
    PIE1bits.TMR2IE = 1;
}
```

```
void init_uart(void)
{
    BAUDCONbits.BRG16 = 1;
    TXSTAbits.BRGH = 1; // p204
    TXSTAbits.SYNC = 0; // p207

    // p207
    SPBRG = 207;
    SPBRGH = 0;

    TRISCbits.RC6 = 1;
    TRISCbits.TRISC6 = 1; // TX1 en entree
    TRISCbits.TRISC7 = 1; // RX1 en entree
    RCSTAbits.SPEN = 1;   // Port srie activ (p205)
    TXSTAbits.TXEN = 1;   // Activation transmission (p2
    RCSTAbits.CREN = 1;   // Activation rception continu
}

void init_I2C(void)…

void init_ADC(void)…

void init_timer(void) // Timer0
{
    // p125
    T0CONbits.T08BIT = 0; // timer en 16 bits
    T0CONbits.T0CS = 0;   // horloge interne
    T0CONbits.T0PS = 1;   // PRE = 1:4

    T0CONbits.PSA = 0;      // L'horloge du Timer0 vient
    INTCONbits.TMR0IE = 1; // on active l'IT overflow du

    TMR0H = 0xD8; // B1 bits de poids fort
    TMR0L = 0xEF; // DF bits de poids faible

    T0CONbits.TMR0ON = 1; // on active le Timer0
}

void valider_IT(void)
{

    INTCONbits.INT0IE = 1;
    INTCON2bits.INTEDG0 = 0; // front descendant
    INTCONbits.GIE = 1;      // on active les IT globale

}
```

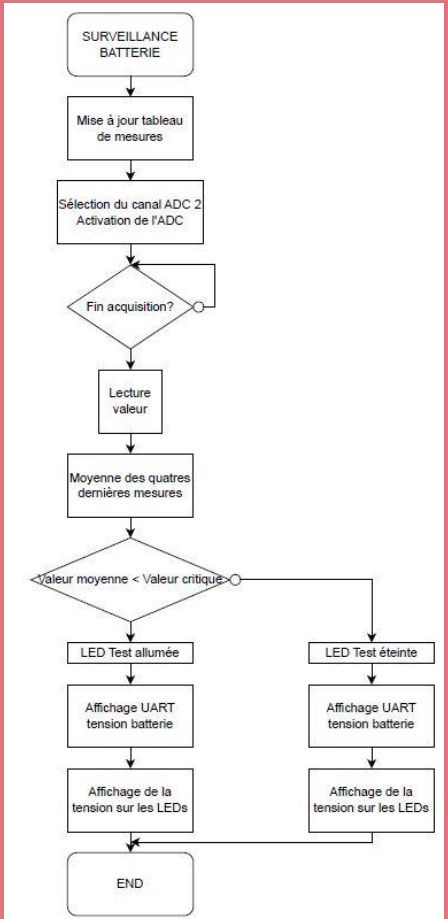
Algorigramme



Code

```c
#pragma code HighVector = 0x08

void IntHighVector(void)
{
    _asm goto HighISR _endasm
}

#pragma code
#pragma interrupt HighISR

void HighISR(void)
{
    //printf("Test dans HighISR\n\r");

    // IT telecommande (Int0)
    if (INTCONbits.INT0IF)
    {
        INTCONbits.INT0IF = 0; // IT acquittée
        flag_telecommande = 1;
        //printf("IT Telecommande\n\r"); // fonctionne
    }

    // IT Timer0
    if (INTCONbits.TMR0IF)
    {
        INTCONbits.TMR0IF = 0; // IT acquittée
        flag_capteurs_IR = 1;
        //printf("IT Timer0\n\r"); // fonctionne
        compteur++;

        // p.17r pour les valeurs
        TMR0H = 0xB1;
        TMR0L = 0xDF;

        if (compteur == 1) // 10 // le compteur se déclen
        {
            flag_batterie = 1;
            compteur = 0;
            //printf("IT Batterie\n\r"); // fonctionne
        }
    }
}
```

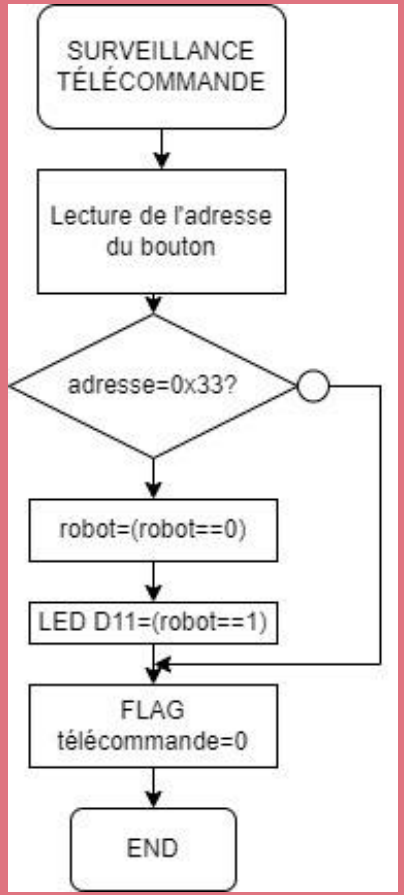
Algorigramme



Code

```
105 void surveillance_batt(void)
106 {
107     // Decalage des indices
108     int i;
109     int val_moy;
110 
111     // décalage des valeurs vers la droite
112     for (i = NB_MESURES_BATT-1; i>0; i--)
113     {
114         liste_UBAT[i] = liste_UBAT[i-1];
115     }
116 
117     // Selection du channel 2
118     ADCON0bits.CHS0 = 0;
119     ADCON0bits.CHS1 = 1;
120     ADCON0bits.CHS2 = 0;
121     ADCON0bits.CHS3 = 0;
122 
123     // Demarrage de la conversion
124     ADCON0bits.GO = 1;
125 
126     while (ADCON0bits.DONE);
127 
128     // Les registres ADRESH et ADRESL contiennent le résultat de la conversion AN
129     UBAT = (unsigned int)ADRESH;
130 
131     liste_UBAT[0] = UBAT;
132 
133     val_moy = moyenne(liste_UBAT, NB_MESURES_BATT); // 4 mesures/seconde ?
134 
135     if (val_moy < VALEUR_TENSION_CRITIQUE)
136     {
137         PORTBbits.RB5 = 1;  // on allume la LED Test
138         printf("Valeur batterie : %d (batterie faible)\r\n", val_moy);
139         affichage_batterie(val_moy);
140     }
141     else
142     {
143         PORTBbits.RB5 = 0;  // on allume la LED Test
144         printf("\r\nValeur batterie : %d \r\n", val_moy);
145         affichage_batterie(val_moy);
146     }
147 }
148 
```

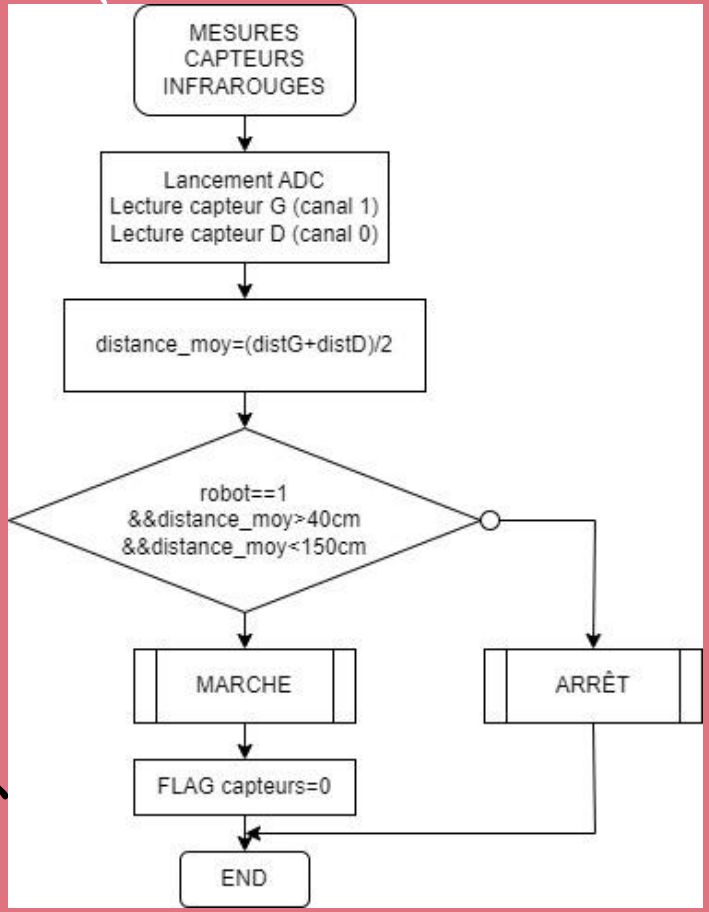
Algorigramme



Code

```
92  void surveillance_tele(void)
93  {
94      char buffer_telec[3]; // tableau utilisé pour stocker les donnees lues
95
96      Lire_i2c_Telecom(ADRESSE_TELEC, buffer_telec); // lecture des donnees et
97
98      if (buffer_telec[1] == ADRESSE_BOUTON) // on verifie que c'est la touche
99      {
100         robot = (robot == 0);
101         PORTBbits.RB5 = (robot == 1);
102     }
103 }
```

Algorigramme



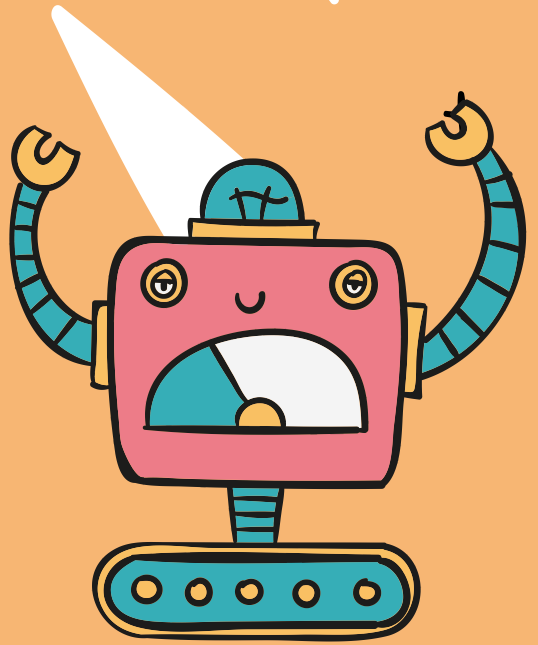
Code

```
50  void surveillance_capt(void)
51  {
52      int val_moy; // utile pour stocker la moyenne des distances
53      int i;
54
55      // décalage des valeurs vers la droite
56      for (i = NB_MESURES_IR-1; i>0; i--)
57      {
58          liste_dist[i] = liste_dist[i-1];
59      }
60
61      // capteur IR droit
62      ADCON0bits.CHS = 0; // channel 0
63      ADCON0bits.GO = 1;
64      while (ADCON0bits.DONE);
65
66      dist_D = (int)ADRESH;
67
68      // capteur IR gauche
69      ADCON0bits.CHS = 1; // channel 1
70      ADCON0bits.GO = 1;
71      while (ADCON0bits.DONE);
72
73      dist_G = (int)ADRESH;
74
75      dist = (dist_G + dist_D) / 2; // moyenne des deux capteurs
76      liste_dist[0] = dist;
77
78      val_moy = moyenne(liste_dist, NB_MESURES_IR); // moyenne des 8 mesures
79
80      printf("dist_D : %d \r\ndist_G : %d\r\n\r\n", dist_D, dist_G);
81
82      if (robot == 1 && val_moy > VALEUR_DISTANCE_IR_MAX && val_moy < VALEUR_DISTANCE_IR_MIN)
83      {
84          vitesse_moteur(PWM_D, PWM_G);
85      }
86      else
87      {
88          vitesse_moteur(STOP_MOTEURS, STOP_MOTEURS);
89      }
90  }
```

03.

Resultats & ameliorations

Résultats des simulations,
améliorations éventuelles et difficultés



I2C

À l'oscilloscope, on observe la liaison I2C qui permet l'affichage LED. Sur la sonde J24, on a SCL (voie 1) et sur la sonde J25 on a SDA (voie 2). En l'absence de réception d'un signal de la télécommande, le microcontrôleur envoie régulièrement une valeur sur U8. En effet, on fait appel à la fonction affichage_batterie après avoir effectuer une mesure de la tension batterie qui envoie une valeur correspondant à l'image de la tension mesurée. Cette valeur permet de régler l'affichage des 8 LEDs. On observe la trame suivante en binaire : 01000 0000 0 1100 0000. Les 8 premiers bits correspondent à l'adresse de U8 : 0x40. Le neuvième bit indique une écriture. Les 8 bits restants indique que le pourcentage de la tension batterie est entre 63% et 75%, ainsi les deux premières LEDs sont éteintes, les six autre sont allumées.

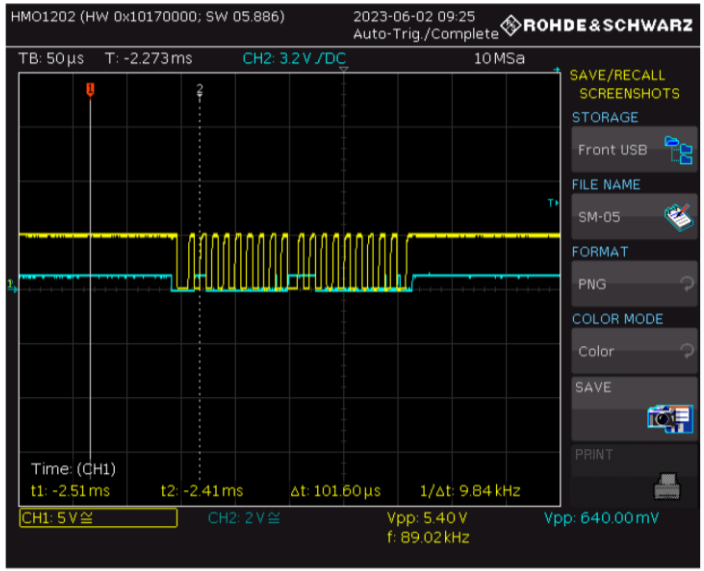


FIGURE 8 – signal I2C observé en J24 (SCL) et J25 (SDA)

Lecture capteurs IR

Au niveau hardware, on observe bien en sortie du capteur une tension analogique. Pour un objet placé à distance fixe du robot, cette tension est fixe. La datasheet du capteur donne une tension d'environ 1.5V pour un objet placé à 40 cm. On retrouve bien cette tension à l'oscilloscope. Au niveau software, on affiche grâce à la liaison UART la valeur numérique correspondante sur les deux capteurs. Comme les capteurs sont légèrement tournés, cette valeur numérique est proche mais pas totalement identique.

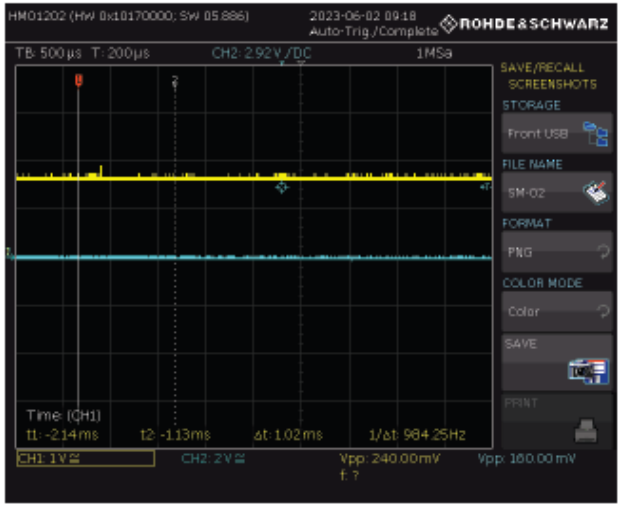


FIGURE 10 – tension en sortie du capteur droit pour un objet à 40 cm

Generation PWM

Dans le cas où les moteurs sont à l'arrêt, le signal PWM a un rapport cyclique RC=0%. On observe alors du bruit d'amplitude de quelques mV sur les sondes J17 et J18. Lorsque le robot est en marche, on observe à la fois un signal PWM du côté du moteur droit et du moteur gauche. Le signal PWM a dans les deux cas un rapport cyclique RC=50%. Le signal PWM a une fréquence de 4000Hz. On a ci-dessous une mesure du signal PWM à l'oscilloscope :

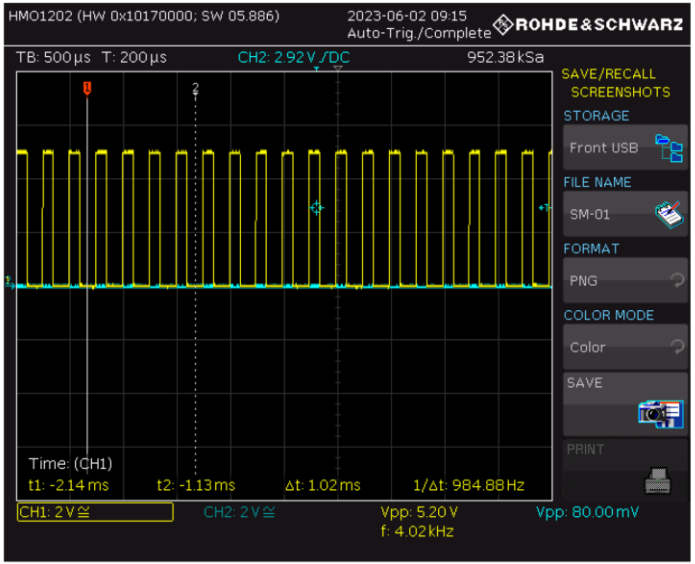


FIGURE 11 – le signal PWM observé sur la sonde J17 (moteur droit)

Difficulte : Capteurs IR

5.2.2 Lecture capteurs IR

Au niveau hardware, on observe bien en sortie du capteur une tension analogique. Pour un objet placé à distance fixe du robot, cette tension est fixe. La datasheet du capteur donne une tension d'environ 1.5V pour un objet placé à 40 cm. On retrouve bien cette tension à l'oscilloscope. Au niveau software, on affiche grâce à la liaison UART la valeur numérique correspondante sur les deux capteurs. Comme les capteurs sont légèrement tournés, cette valeur numérique est proche mais pas totalement identique.

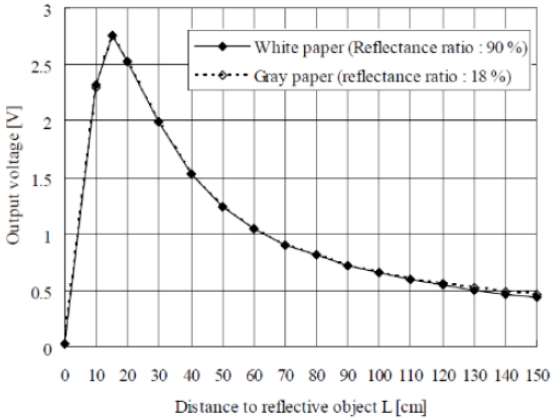


FIGURE 9 – extrait datasheet capteurs IR

Pistes d'amelioration

Utilisation d'un testeur de batterie : Un testeur de batterie mesure la charge restante dans une batterie. Il peut donner une indication plus précise de l'état de la batterie que l'estimation basée uniquement sur la tension. C'est particulièrement utile dans des systèmes où le niveau de la batterie peut avoir un impact significatif sur le fonctionnement.

Utilisation d'un bipeur en cas de tension critique : Un bipeur produit un signal sonore lorsqu'il est activé. Ici, il serait utilisé pour alerter l'utilisateur lorsque la tension de la batterie tombe en dessous d'un certain seuil critique (10V). Cette amélioration permet une réaction rapide à un état de batterie faible, qui pourrait passer inaperçu.

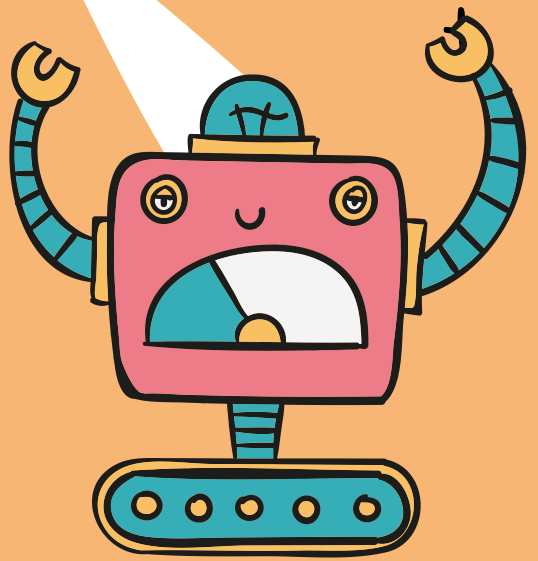
Utilisation d'une batterie lithium polymère 3 cellules : Les batteries au lithium polymère (ou LiPo) sont des types de batteries rechargeables qui offrent plusieurs avantages par rapport aux autres types de batteries, y compris une densité d'énergie plus élevée, un taux de décharge plus élevé, et la possibilité de prendre presque n'importe quelle forme. L'utilisation d'une batterie LiPo 3 cellules fournirait une tension nominale de 11.1V, ce qui serait suffisant pour alimenter le système en 12V.

Utilisation d'un testeur de batterie qui déclenche un signal sonore en cas de sous-alimentation : Comme mentionné précédemment, un testeur de batterie peut fournir une indication précise de l'état de la batterie. Cependant, ce testeur particulier serait également capable de déclencher un signal sonore si la tension de la batterie tombe en dessous de 10V. Cela offrirait un double avantage : non seulement il donnerait une indication précise de l'état de la batterie, mais il alerterait également l'utilisateur de toute sous-alimentation, permettant une intervention rapide.

04.

Conclusion

Réussite et bilan d'expérience
(ce que nous avons obtenu, retenu et apprécié)



PROJET ROBOT 2

Soutenance

Théo GACHET Cyril HÉRAIL