

**B. DHALLUIN**

**JP. BOURGUET**

**RAPPORT PROJET ROBOT**

Design logiciel et programmation

Date : 17/03/2016

**EMSE**



Cursus ISMIN

Gestion Documentaire

|  |
| --- |
| **OBJECTIFS DU DOCUMENT** |
| Ce document a pour but d’exposer les étapes et avancées effectuées au cours des diverses séances sur les choix techniques et l’élaboration des différentes fonctionnalités du robot imposées par le cahier des charges fonctionnel établi par notre contrat. |

|  |  |
| --- | --- |
| **ETAPES D’ELABORATION** | **Date** |
| S.A.M. 1 | Semestre 1 |
| S.A.M. 2. : Projet Pilotage Robot | Semestre 2 partie 1 |
| Projet Robot 2 | Semestre 2 partie 2 |
| Elaboration d’algorigramme |  |
| Implémentation des programmes sous MPLAB X IDE |  |
| Affinage des programmes sous ISIS |  |
| Tests sur robot pour tenir compte des contraintes mécaniques et physiques |  |

|  |  |
| --- | --- |
| **DEMANDANT** | **DESTINATAIRES(S)** |
| E.M.S.E. – cursus I.S.M.I.N. – Projet Robot | B. Dhalluin |
| E.M.S.E. – cursus I.S.M.I.N. – Projet Robot | JP. Bourget |

Sommaire

[1. Présentation du sujet 4](#_Toc453880521)

[2. Notre contrat ; LE CONTRAT 6 4](#_Toc453880522)

[3. Problèmes rencontrés 5](#_Toc453880523)

[4. Respect des consignes et bilan qualitatif 5](#_Toc453880524)

[5. Annexes 6](#_Toc453880525)

[5.1 Init.c 6](#_Toc453880526)

[5.2 Main.c 9](#_Toc453880527)

[5.3 Variables.h 17](#_Toc453880528)

# Présentation du sujet

Le projet robot permet de mettre en œuvre toutes les compétences nécessaires lors de la réalisation d’un projet de conception pour un système embarqué. Durant les deux semestres nous avons travaillé sur le robot avec toute la réalisation de la carte et ensuite le développement et les tests de notre code le robot. Mais la réalisation de notre robot aurait été impossible si nous n’avions pas eu de cours de Système A Microcontrôleur, ces cours nous ont permis de nous introduire à l’assembleur pour ensuite être capable de réaliser nous-même toutes les fonctionnalités de nôtre robot.

# Notre contrat ; LE CONTRAT 6

En premier Le contrat qui nous a été attribué est le numéro 6, il reprend le principe général d’une voiture téléguidée. Les objectifs de navigation du contrat sont les suivant. Le robot devra réagir aux 5 Boutons de la télécommande. Les boutons Avance et Recul modifient la vitesse moyenne et les boutons Droite et Gauche feront tourner vers la droite ou la gauche le robot.

Deux niveaux doivent être pris en compte pour les variations de vitesse ou d'angle. C'est à dire :

• Bouton Avance : Vitesse moyenne augmente vers l'avant : 2 valeurs 7 ou 12 m/mn (± 20%),

• Bouton Recul : Vitesse moyenne augmente vers l'arrière : 2 valeurs 7 ou 12 m/mn (± 20%),

• Bouton Droite : Virage à droite, 2 valeurs d'angle, de braquage : 1,5m de rayon (± 50%), 0,5m de rayon (± 50%). Que l'on avance ou que l'on recule : pas de changement de vitesse moyenne.

• Bouton Gauche : Virage à gauche, 2 valeurs d'angle, de braquage : 1,5m de rayon (± 50%), 0,5m de rayon (± 50%). Que l'on avance ou que l'on recule : pas de changement de vitesse moyenne.

• Bouton Central = Arrêt et annulation d'un ordre de braquage précédent.

Une nouvelle commande contradictoire annule sa réciproque : En cas d'avance, l'appui sur recul produit l'arrêt (sans changer de direction), et inversement En cas de virage à droite, l'appui sur le bouton Gauche provoque la mise en marche (avant ou arrière) tout droit, et inversement.

Les performances attendues sont évaluées en testant chacun des termes du contrat ci-dessus. En braquage nul, on ne s'intéressera pas au décalage de vitesses des moteurs D/G inhérents aux frottements différents des chaînes mécaniques. Il faudra montrer que les signaux PWM sont identiques dans tous les cas.

On emploiera une télécommande identifiée permanente pour ce contrat. Amélioration proposée : La mise en place d’un demi-tour (lorsque le robot est à l’arrêt, si on appuie sur la flèche droite ou gauche il effectue un demi-tour qui doit être arrêté par l’utilisateur) géré par l’utilisateur via la flèche droite et gauche lorsque nous sommes à l’arrêt.

# Elaborations et différentes fonctions

# D:\PC_Esteban\ISMIN\ProjetRobot\projetRobot-main.pngGraphe d’états et diagramme

# Problèmes rencontrés

# Respect des consignes et bilan qualitatif

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 |
| 2 vitesses longitudinales |  |  |  | **X** |
| 2 rayons  braquage |  |  |  | **X** |
| Bouton d’arrêt d’urgence |  |  |  | **X** |
| Rotation sur place |  | **X** |  |  |
|  |  |  |  |  |

# Annexes

## Init.c

1. #include<p18f2520.h>
2. #include"init.h"
3. #include "variablesExt.h"
5. void init(void){
6. */\* Initialisation Horloge* *\*/*
7. OSCCONbits.IRCF2 = 1;   *//Horloge de 8MHz*
8. OSCCONbits.IRCF1 = 1;
9. OSCCONbits.IRCF0 = 1;
11. */\* Initialisation Ports et Pins* *\*/*
12. TRISCbits.RC4 = 1;        *//Broche SDA en entree*
13. TRISCbits.RC3 = 1;        *//Broche SCL en entree*
15. TRISCbits.RC1 = 1;        *//Initialisation PWM droit*
16. TRISCbits.RC2 = 1;        *//Initialisation PWM gauche*
18. TRISAbits.RA6 = 0;        *//Port RA6 en sortie*
19. TRISAbits.RA7 = 0;        *//Port RA7 en sortie*
20. TRISBbits.RB1 = 0;        *//Port RB1 en sortie*
22. TRISCbits.RC6 = 1;        *//Initialisation TX*
23. TRISCbits.RC7 = 0;        *//Initialisation RX*
25. */\* Initialisation des Périphériques*  *\*/*
26. *// Initialisation I2C*
27. MI2CInit();
29. *// Initialisation RS232*
30. SPBRG = 25;                 *//Initilisation Baudrate*
31. TXSTAbits.BRGH = 0;
32. TXSTAbits.SYNC = 1;
33. BAUDCONbits.BRG16 = 0;
34. RCSTAbits.SPEN = 1;         *//Validation TX*
35. TXSTAbits.TXEN = 1;         *//Vaidation transmission*
37. *// Initialisation ADC*
38. *//Initialisation en mod analogique et selection Vss et vdd*
39. ADCON1 = 0;
40. ADCON1bits.PCFG2 = 1;
41. ADCON1bits.PCFG3 = 1;
43. *//Choix Tad et Taco et calage a gauche*
44. ADCON2 = 0;
45. ADCON2bits.ACQT1 = 1;
46. ADCON2bits.ACQT0 = 1;
47. ADCON2bits.ADCS2 = 1;
49. *//AN2 en entr�e analogique*
50. ADCON0bits.CHS1 = 1;
52. ADCON0bits.ADON = 1;
54. *// Initialisation PWM*
55. TRISCbits.RC2 = 0; *// ccp1 sortie*
56. TRISCbits.RC1 = 0; *// ccp2 sortie*
57. TRISAbits.RA6 = 0; *// configurer en sortie*
58. TRISBbits.RB0=1;   *// configurer en entree  relie a l interruption INT0 (liaison niveau materiel)*
59. TRISBbits.RB5=0;   *// configurer en sortie led*
60. TRISBbits.RB1=1;  *// int 1 bouton*
62. T2CONbits.T2CKPS1 = 0;
63. T2CONbits.T2CKPS0 = 1; *// on met le prescaleur a 4 on a PR2 = 249*
65. PR2 = 249;  *//  choix PWM*
67. *// CCPRxL:DC1B1:DC1B0 = 200 donc CCPRxL = 200 / 4*
68. CCPR1L = 0;*//50 ;                        //50*
69. CCP1CONbits.DC1B1 = 0;               *//0*
70. CCP1CONbits.DC1B0 = 0;               *//0*
72. CCPR2L = 0;*//50 ;                        //50*
73. CCP2CONbits.DC2B1 = 0;               *//0*
74. CCP2CONbits.DC2B0 = 0;
76. T2CONbits.TMR2ON = 1; *// active timer*
78. CCP1CONbits.CCP1M3 = 1; *// activation du mode PWM*
79. CCP1CONbits.CCP1M2 = 1;
80. CCP2CONbits.CCP2M3 = 1; *// activation du mode PWM*
81. CCP2CONbits.CCP2M2 = 1;
83. T2CONbits.T2OUTPS3 = 1; *// gestion postscaleur pour base de temps*
84. T2CONbits.T2OUTPS2 = 0;  *// on divise par 10 pour passer de 1ms a 10ms*
85. T2CONbits.T2OUTPS1 = 1;  *// de periode*
86. T2CONbits.T2OUTPS0 = 0;
88. *// Initialisation Interrupt*
89. INTCON2bits.INTEDG0 = 0;    *//Interruption sur front descendant*
90. INTCONbits.INT0IE = 1;      *//interruption 0*
91. INTCONbits.INT0IF = 0;
92. PIE1bits.TMR2IE = 1; *//Active l'interruption TIMER2*
93. INTCONbits.PEIE = 1;        *//Validation interruptions*
94. INTCONbits.GIE = 1;         *//Validation interruptions*
95. c10msTel = 0;
96. return ;
97. }

## Main.c

1. #include "initASM.h"
2. #include "variables.h"
3. #include "MI2C.h"
4. #include "init.h"
5. #include "main.h"
7. #pragma interrupt HighISR
8. void HighISR(void)
9. {
10. if(PIR1bits.TMR2IF){*//si timer interrupt*
11. PIR1bits.TMR2IF = 0;
12. c10msTel++;
13. if(c100ms++ > 10)
14. flag\_100ms = 1;
15. if(c2s++ > 200)
16. flag\_2s = 1;
17. }
18. if (INTCONbits.INT0IF){
19. INTCONbits.INT0IF = 0;
20. flag\_telecom = 1;
21. }
22. }
23. char datasToChart(datas d){
24. char c;
25. c = 0;
26. c |= d.actuel.vitesse;
27. c |= d.actuel.avantArriere <<2;
28. c |= d.actuel.gaucheDroite <<4;
29. c |= d.actuel.angle<<6;
31. return c;
32. }
34. void saveState(datas \*etat){
35. etat->precedent.angle           = etat->actuel.angle;
36. etat->precedent.avantArriere    = etat->actuel.avantArriere;
37. etat->precedent.gaucheDroite    = etat->actuel.gaucheDroite;
38. etat->precedent.vitesse         = etat->actuel.vitesse;
39. }
41. void gestionBatt(){
42. c2s = 0; *// reset compteur*
43. flag\_2s = 0; *//reset flag*
44. ADCON0bits.GO = 1; *// d�but conv*
45. while(!ADCON0bits.DONE); *//attend fin conv*
46. mi[mesure] = ADRESH;
47. mesure++;
48. mesure = mesure%4;
49. if (mi[3] > 0){
50. Vbat = (mi[0]+mi[1]+mi[2]+mi[3])/4;
51. if (Vbat < 169){*//SeuilBat){ //Si sous tension de seuil 10V*
52. PORTBbits.RB5=0;
53. }
54. else{
55. PORTBbits.RB5=1;
56. }
57. }
58. }
60. datas actualiseTelecom(datas etat){
61. Lire\_i2c\_Telecom(0xA2, lectureTelecom);
62. saveState(&etat);
63. switch(lectureTelecom[1]){
64. case 0x31:
65. if(etat.precedent.gaucheDroite==0 && etat.precedent.angle==0)
66. {
67. etat.actuel.gaucheDroite=1;
68. etat.actuel.angle=1;
69. }
70. else if(etat.precedent.gaucheDroite==1 )
71. {
72. etat.actuel.gaucheDroite=1;
73. etat.actuel.angle=2;
74. }
75. else if(etat.precedent.gaucheDroite==2 && etat.precedent.angle==2)
76. {
77. etat.actuel.gaucheDroite=2;
78. etat.actuel.angle=1;
79. }
80. else if(etat.precedent.gaucheDroite==2 && etat.precedent.angle==1)
81. {
82. etat.actuel.gaucheDroite=0;
83. etat.actuel.angle=0;
84. }
86. **break**;
87. case 0x32:
88. if (etat.precedent.avantArriere==0
89. && etat.precedent.vitesse==0){
90. etat.actuel.vitesse = 1;
91. etat.actuel.avantArriere = 1;
92. }
93. else if (etat.precedent.avantArriere==1
94. && etat.precedent.vitesse==1){
95. etat.actuel.vitesse = 2;
96. }
97. else if (etat.precedent.avantArriere==2
98. && etat.precedent.vitesse==2){
99. etat.actuel.vitesse = 1;
100. }
101. else if (etat.precedent.avantArriere==2
102. && etat.precedent.vitesse==1){
103. etat.actuel.vitesse = 0;
104. etat.actuel.avantArriere=0;
105. }
106. **break**;
107. case 0x33:
108. flag\_IsOn = 1-flag\_IsOn;
109. if (flag\_IsOn == 0){
110. etat.actuel.vitesse = 0;
111. etat.actuel.angle=0;
112. etat.actuel.avantArriere = 0;
113. etat.actuel.gaucheDroite = 0;
114. }
116. **break**;
117. case 0x34:
118. if (etat.precedent.avantArriere==0
119. && etat.precedent.vitesse==0){
120. etat.actuel.vitesse = 1;
121. etat.actuel.avantArriere = 2;
122. }
123. else if (etat.precedent.avantArriere==2
124. && etat.precedent.vitesse==1){
125. etat.actuel.vitesse = 2;
126. }
127. else if (etat.precedent.avantArriere==1
128. && etat.precedent.vitesse==2){
129. etat.actuel.vitesse = 1;
130. }
131. else if (etat.precedent.avantArriere==1
132. && etat.precedent.vitesse==1){
133. etat.actuel.vitesse = 0;
134. etat.actuel.avantArriere=0;
135. }
136. **break**;
137. case 0x35:
138. if(etat.precedent.gaucheDroite==0 && etat.precedent.angle==0)
139. {
140. etat.actuel.gaucheDroite=2;
141. etat.actuel.angle=1;
142. }
143. else if(etat.precedent.gaucheDroite==2 )
144. {
145. etat.actuel.gaucheDroite=2;
146. etat.actuel.angle=2;
147. }
148. else if(etat.precedent.gaucheDroite==1 && etat.precedent.angle==2)
149. {
150. etat.actuel.gaucheDroite=1;
151. etat.actuel.angle=1;
152. }
153. else if(etat.precedent.gaucheDroite==1 && etat.precedent.angle==1)
154. {
155. etat.actuel.gaucheDroite=0;
156. etat.actuel.angle=0;
157. }
158. **break**;
159. default:
160. **break**;
161. }
162. lectureTelecom[1] = 0;
163. Write\_PCF8574(0x40, ~datasToChart(etat));
164. return etat;
165. }
167. void afficherRS232(){
168. return;
169. }
171. datas navigue(datas etats){*//\**
172. switch(etats.actuel.avantArriere){
173. case 0:
174. CCPR1L = 0; CCPR2L = 0;
175. **break**;
176. case 1:
177. PORTAbits.RA6 = 1; PORTAbits.RA7 = 1;
178. **break**;
179. case 2:
180. PORTAbits.RA6 = 0; PORTAbits.RA7 = 0;
181. **break**;
182. default:
183. Write\_PCF8574(0x40, ~0xFF);
184. **break**;
185. }
186. if (etats.actuel.avantArriere!=0){
187. switch(etats.actuel.vitesse){
188. case 0:
189. CCPR1L = 0; CCPR2L = 0;
190. **break**;
191. case 1:
192. CCPR1L = v1 ; CCPR2L = v1;
193. **break**;
194. case 2:
195. CCPR1L = v2; CCPR2L = v2;
196. **break**;
197. default:
198. CCPR1L = 0; CCPR2L = 0;
199. **break**;
200. }
201. if (etats.actuel.gaucheDroite == 0){
202. CCPR1L = CCPR2L;
203. }
204. else if (etats.actuel.gaucheDroite!=0){
205. switch(etats.actuel.angle){
206. case 0:
207. CCPR2L = CCPR1L;
208. **break**;
209. case 2:
210. if (etats.actuel.gaucheDroite == 2){
211. CCPR1L = 70 ; CCPR2L = 150;
212. }
213. else if (etats.actuel.gaucheDroite == 1){
214. CCPR1L = 150; CCPR2L = 70 ;
215. }
216. **break**;
217. case 1:
218. if (etats.actuel.gaucheDroite == 2){
219. CCPR1L = 100; CCPR2L = 150;
220. }
221. else if (etats.actuel.gaucheDroite == 1){
222. CCPR1L = 150; CCPR2L = 100;
223. }
224. **break**;
225. default:
226. CCPR1L = 0; CCPR2L = 0;
227. **break**;
228. }
229. }
230. }
231. return etats;
232. }
233. void stop(){
234. saveState(&donnees);
235. donnees.actuel.angle=0;
236. donnees.actuel.avantArriere = 0;
237. donnees.actuel.gaucheDroite = 0;
238. donnees.actuel.vitesse = 0;
239. CCPR1L = 00 ;
240. CCPR2L = 00 ;
241. return;
242. }
243. void main(void){
244. char c;
245. init();
246. flag\_IsOn = 0;
247. donnees.precedent.angle=0;
248. donnees.precedent.avantArriere = 0;
249. donnees.precedent.gaucheDroite = 0;
250. donnees.precedent.vitesse = 0;
251. donnees.actuel.angle=0;
252. donnees.actuel.avantArriere = 0;
253. donnees.actuel.gaucheDroite = 0;
254. donnees.actuel.vitesse = 0;
255. Write\_PCF8574(0x40, ~datasToChart(donnees));
256. CCPR1L = 0;
257. CCPR2L = 0;
258. while(1){
259. afficherRS232();
260. if(flag\_2s){
261. gestionBatt();
262. }
263. if(flag\_telecom){
264. flag\_telecom = 0;
265. if (c10msTel > 5){
266. c10msTel = 0;
267. donnees = actualiseTelecom(donnees);
268. }
269. }
270. if(flag\_IsOn == 1){
271. donnees = navigue(donnees);
272. }
273. else{
274. stop();
275. }
276. }
277. }

## Variables.h

1. #include <stdio.h>
2. #include <stdlib.h>
3. #define SeuilBat 169
4. #define v1 50
5. #define v2 100
7. typedef struct{
8. *//avantArriere = 1 => avant //gaucheDroite = 1 => droite*
9. *//avantArriere = 2 => arri�re //gaucheDroite = 2 => gauche*
10. *//avantArriere = 0 => point mort //gaucheDroite = 0 => ligne droite*
11. unsigned  avantArriere:2; unsigned  gaucheDroite:2;
12. unsigned  vitesse:2; unsigned  angle:2;
13. }etat;
14. typedef struct{
15. etat actuel;
16. etat precedent;
17. }datas;
19. */\* Variables relatives aux interruptions \*/*
20. unsigned char flag\_telecom;
21. unsigned char flag\_100ms;
22. unsigned char flag\_2s;
23. unsigned char c100ms;
24. unsigned char c10msTel;*//\*/*
25. unsigned char c2s;
26. unsigned char flag\_IsOn;
28. */\* Variables relatives a la gestion de la batterie \*/*
29. unsigned int mi[] = {0,0,0,0};
30. unsigned char mesure;
31. unsigned char Vbat;
32. char lectureTelecom[3];
33. datas donnees;