

# PROJET EVABOT AUTOBOT

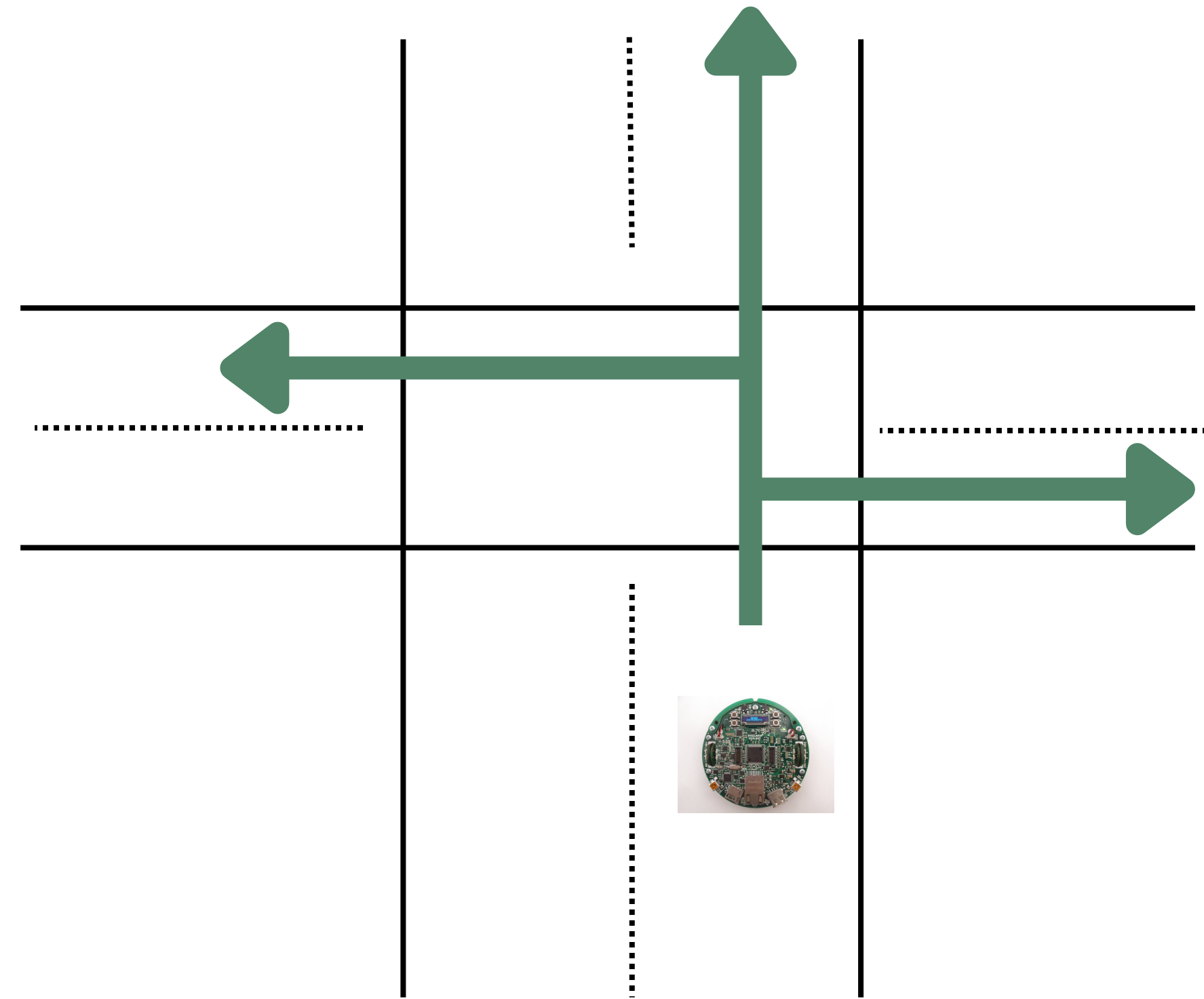


JABRY RANA  
WANDAOGO ABIBATOU

# DESCRIPTION DU PROJET :

## Comportement du robot

- La voiture avance tout droit.
- Si un obstacle est détecté par l'un des deux **bumpers**, elle s'arrête pendant à peu près 3 secondes, puis reprend son chemin en avant.
- Lorsque le **switch 1** est activé, la voiture tourne à droite et la **LED droite** s'allume.
- Lorsque le **switch 2** est activé, la voiture tourne à gauche et la **LED gauche** s'allume.



Simulation d'une voiture intelligente

# CONFIGURATION GPIO :

GPIO_PORTF_BASE	0x40025000
GPIO_PORTE_BASE	0x40024000
GPIO_PORTD_BASE	0x40007000
BROCHE4	Led 1
BROCHE5	Led 2
BROCHE0	Bumper 1
BROCHE1	Bumper 2
BROCHE6	Switch 1
BROCHE7	Switch 2

```
10 GPIO_PORTF_BASE EQU 0x40025000
11
12 GPIO_PORTE_BASE EQU 0x40024000
13
14 GPIO_PORTD_BASE EQU 0x40007000
15
16 ;-----
17
18 GPIO_O_DIR EQU 0x00000400
19
20 GPIO_O_DR2R EQU 0x00000500
21
22 GPIO_O_DEN EQU 0x0000051C
23
24 GPIO_I_PUR EQU 0x00000510
25
26 ;-----
27
28 BROCHE4 EQU 0x10 ; led1 sur broche 4
29 BROCHE5 EQU 0x20 ; led2 sur broche 5
30 BROCHE0 EQU 0x01 ; bumper 1
31 BROCHE1 EQU 0x02 ; bumper 2
32 BROCHE6 EQU 0x40 ; bouton poussoir 1
33 BROCHE7 EQU 0x80 ; bouton poussoir 2
34
```

# DÉROULEMENT DU PROGRAMME :

## Configuration des Leds, Switch, et Bumpers

```
;/*****CONFIGURATION LED 1 et 2

ldr r6, = GPIO_PORTF_BASE+GPIO_O_DIR    ;; 1 Pin du portF en sortie (broche 4 : 00010000)
ldr r0, = BROCHE4 + BROCHE5
str r0, [r6]

ldr r6, = GPIO_PORTF_BASE+GPIO_O_DEN    ;; Enable Digital Function
ldr r0, = BROCHE4 + BROCHE5
str r0, [r6]

ldr r6, = GPIO_PORTF_BASE+GPIO_O_DR2R   ;; Choix de l'intensité de sortie (2mA)
ldr r0, = BROCHE4 + BROCHE5
str r0, [r6]

mov r2, #0x000                          ;; pour eteindre LED

; allumer la led broche 4 (BROCHE4_5)
mov r3, #BROCHE4                        ;; Allume LED1&2 portF broche 4&5 : 00110000
mov r4, #BROCHE5

ldr r7, = GPIO_PORTF_BASE + (BROCHE4<<2) ;; @data Register = @base + (mask<<2) ==> LED1
ldr r8, = GPIO_PORTF_BASE + (BROCHE5<<2)
;*****Fin configuration LED
```

```
;/*****CONFIGURATION Switch 1 & 2

ldr r9, = GPIO_PORTD_BASE+GPIO_I_PUR    ;; Pul_up
ldr r0, = BROCHE6 + BROCHE7
str r0, [r9]

ldr r9, = GPIO_PORTD_BASE+GPIO_O_DEN    ;; Enable Digital Function
ldr r0, = BROCHE6 + BROCHE7
str r0, [r9]

ldr r11, = GPIO_PORTD_BASE + (BROCHE6<<2)
ldr r12, = GPIO_PORTD_BASE + (BROCHE7<<2)
;*****Fin configuration Bumper
```

```
;/*****CONFIGURATION Bumper 1 & 2

ldr r9, = GPIO_PORTC_BASE+GPIO_I_PUR    ;; Pul_up
ldr r0, = BROCHE0 + BROCHE1
str r0, [r9]

ldr r9, = GPIO_PORTC_BASE+GPIO_O_DEN    ;; Enable Digital Function
ldr r0, = BROCHE0 + BROCHE1
str r0, [r9]

ldr r9, = GPIO_PORTC_BASE + (BROCHE0<<2)
ldr r10, = GPIO_PORTC_BASE + (BROCHE1<<2)
;*****Fin configuration Bumper
```

# DÉROULEMENT DU PROGRAMME :

## Démarrage du Robot et vérification des switchs et bumpers

```
    ; Configure les PWM + GPIO
BL  MOTEUR_INIT
BL  MARCHE
    ; Activer les deux moteurs droit et gauche

////////////////////////////////////

MARCHE
    BL  MOTEUR_DROIT_ON
    BL  MOTEUR_GAUCHE_ON
    BL  MOTEUR_DROIT_AVANT
    BL  MOTEUR_GAUCHE_AVANT    ;; Activer le moteur et le Robot Avance
    BL  ALLUMER_LED1
    BL  ALLUMER_LED2          ;; Allumer led1 ou led2
    BL  VERIF BUMPER1          ;; test bumpers 2
    BL  VERIF BUMPER2          ;; test bumpers 1
    BL  VERIF_SWITCH1          ;; verifie switch1
    BL  VERIF_SWITCH2          ;; verifie switch2

B MARCHE    ;; revenir à la marche
```

# DÉROULEMENT DU PROGRAMME :

Test si les bumpers sont appuyés

```
VERIF_BUMPER1
    ldr r0,[r9]           ;; r0 la valeur du bumper 1
    CMP r0,#0x00         ;; test si bumper 1 appuyé
    BEQ ARRETER          ;; arrete le robot

    BX LR                ;; sinon on revient à la ligne suivant du programme ;; on execute VERIF_BUMPER2

VERIF_BUMPER2
    ldr r0,[r10]          ;; r0 la valeur du bumper 2
    CMP r0,#0x00         ;; test si bumper 2 appuyé lorsque r0 égale 0x00
    BEQ ARRETER          ;; arrêter le robot

    BX LR                ;; sinon on revient à la ligne suivant du programme

ARRETER
    BL MOTEUR_GAUCHE_OFF
    BL MOTEUR_DROIT_OFF  ;; désactiver le robot
    BL WAIT_ARRET       ;; pendant une periode
    B MARCHE            ;; puis réactiver le robot
```



# DÉROULEMENT DU PROGRAMME :

## Test si les Switchs sont appuyés

```
VERIF_SWITCH1
    ldr r6,[r11]           ;; r6 la valeur du switch1
    CMP r6,#0x00          ;; test si switch1 appuyer
    BEQ TOURNER_DROIT     ;; le robot tourne à droite
    BX LR                 ;; le robot avance

VERIF_SWITCH2
    ldr r6,[r12]           ;; r6 prend la valeur du switch2
    CMP r6,#0x00          ;; test si switch2 appuyer
    BEQ TOURNER_GAUCHE    ;; si oui , executer le robot tourne à gauche
    BX LR                 ;;le robot avance
```

```
TOURNER_DROIT
    BL ALLUMER_LED1        ;; allumer led1
    BL MOTEUR_DROIT_ARRIERE
    BL WAIT_TOURNER        ;; tourner à droite pendant une periode
    B MARCHE               ;; puis avancer

WAIT_ARRET
    ldr r1, = DUREE_ARRET ;; r1 prend la valeur de durée tourner
wait1 subs r1, #1
    bne wait1 ;;branchement pour tester à nouveau wait1
    BX LR

WAIT_TOURNER    ;; durée pour tourner le robot soit à droite , soit au gauche
    ldr r1, = DUREE_TOURNER ;; r1 prend la valeur de durée tourner
wait2 subs r1, #1
    bne wait2 ;; branchement pour tester à nouveau wait2
    BX LR

TOURNER_GAUCHE
    BL ALLUMER_LED2
    BL MOTEUR_GAUCHE_ARRIERE
    BL WAIT_TOURNER ;; tourne à gauche pendant une periode
    B MARCHE ;; revenir à Marche
```

# Vidéo de notre Robot



# Ouverture

- Intégration de **capteurs** de proximité pour détecter les **obstacles** à distance plutôt que de se baser sur les **bumpers**.
- Utilisation de **caméras** pour détecter les piétons à distance et ajuster la vitesse de la voiture progressivement jusqu'à l'arrêt au passage piéton.

MERCI POUR VOTRE ATTENTION !

# Annexe

## (Code en entier)

```
4      AREA    [.text], CODE, READONLY
5
6      SYSCTL_PERIPH_GPIO EQU    0x400FE108
7
8      GPIO_PORTF_BASE EQU    0x40025000
9
10     GPIO_PORTF_BASE EQU    0x40024000
11
12     GPIO_PORTD_BASE EQU    0x40007000
13
14     -----
15
16     GPIO_O_DIR EQU    0x00000400
17
18     GPIO_O_DR2R EQU    0x00000500
19
20     GPIO_O_DEN EQU    0x0000051C
21
22     GPIO_I_FUR EQU    0x00000510
23
24     -----
25
26     BROCHE4 EQU    0x10 ; led1 sur broche 4
27     BROCHE5 EQU    0x20 ; led2 sur broche 5
28     BROCHE0 EQU    0x01 ; bumper 1
29     BROCHE1 EQU    0x02 ; bumper 2
30     BROCHE6 EQU    0x40 ; bouton poussoir 1
31     BROCHE7 EQU    0x80 ; bouton poussoir 2
32
33     -----
34
35     ; blinking frequency
36     DUREE_TOURNER EQU    0x01303F7F
37     DUREE_ARRET EQU    0x4c4b40
38
39
40
```

```
42     ENTRY    __main
43     EXPORT    __main
44
45     ; The IMPORT command specifies that a symbol is defined in a shared object at runtime.
46     IMPORT    MOTEUR_INIT ; initialise les moteurs (configure les pins + GPIO)
47
48     IMPORT    MOTEUR_DROIT_ON ; activer le moteur droit
49     IMPORT    MOTEUR_DROIT_OFF ; désactiver le moteur droit
50     IMPORT    MOTEUR_DROIT_AVANT ; moteur droit tourne vers l'avant
51     IMPORT    MOTEUR_DROIT_ARRIERE ; moteur droit tourne vers l'arrière
52
53     IMPORT    MOTEUR_GAUCHE_ON ; activer le moteur gauche
54     IMPORT    MOTEUR_GAUCHE_OFF ; désactiver le moteur gauche
55     IMPORT    MOTEUR_GAUCHE_AVANT ; moteur gauche tourne vers l'avant
56     IMPORT    MOTEUR_GAUCHE_ARRIERE ; moteur gauche tourne vers l'arrière
57
58
59
60
61
62     __main
63
64     ldr r6, = SYSCTL_PERIPH_GPIO
65     mov r0, #0x00000038
66     str r0, [r6]
67
68
69     nop
70     nop
71     nop
72
```

```
72     ;*****CONFIGURATION LED 1 et 2
73
74     ldr r6, = GPIO_PORTF_BASE+GPIO_O_DIR ; 1 Pin du portF en sortie (broche 4 : 00010000)
75     ldr r0, = BROCHE4 + BROCHE5
76     str r0, [r6]
77
78
79     ldr r6, = GPIO_PORTF_BASE+GPIO_O_DEN ; Enable Digital Function
80     ldr r0, = BROCHE4 + BROCHE5
81     str r0, [r6]
82
83     ldr r6, = GPIO_PORTF_BASE+GPIO_O_DR2R ; Choix de l'intensité de sortie (2mA)
84     ldr r0, = BROCHE4 + BROCHE5
85     str r0, [r6]
86
87     mov r2, #0x000 ; pour eteindre LED
88
89     ; allumer la led broche 4 (BROCHE4 5)
90     mov r3, #BROCHE4 ; Allume LED1&2 portF broche 4&5 : 00110000
91     mov r4, #BROCHE5
92
93     ldr r7, = GPIO_PORTF_BASE + (BROCHE4<<2) ; @data Register = @base + (mask<<2) ==> LED1
94     ldr r8, = GPIO_PORTF_BASE + (BROCHE5<<2)
95     ;*****Fin configuration LED
96
```

```
96     ;*****CONFIGURATION Switch 1 & 2
97
98     ldr r9, = GPIO_PORTD_BASE+GPIO_I_FUR ; Pul_up
99     ldr r0, = BROCHE6 + BROCHE7
100    str r0, [r9]
101
102
103    ldr r9, = GPIO_PORTD_BASE+GPIO_O_DEN ; Enable Digital Function
104    ldr r0, = BROCHE6 + BROCHE7
105    str r0, [r9]
106
107    ldr r11, = GPIO_PORTD_BASE + (BROCHE6<<2)
108    ldr r12, = GPIO_PORTD_BASE + (BROCHE7<<2)
109    ;*****Fin configuration Bumper
110
```

```
111    ;*****CONFIGURATION Bumper 1 & 2
112
113    ldr r9, = GPIO_PORTF_BASE+GPIO_I_FUR ; Pul_up
114    ldr r0, = BROCHE0 + BROCHE1
115    str r0, [r9]
116
117    ldr r9, = GPIO_PORTF_BASE+GPIO_O_DEN ; Enable Digital Function
118    ldr r0, = BROCHE0 + BROCHE1
119    str r0, [r9]
120
121    ldr r9, = GPIO_PORTF_BASE + (BROCHE0<<2)
122    ldr r10, = GPIO_PORTF_BASE + (BROCHE1<<2)
123    ;*****Fin configuration Bumper
124
```

```
131    ; EL Branchement vers un lien (sous programme)
132    str r9, [r7]
133    str r4, [r8]
134    ; Configure les PWM + GPIO
135    BL MOTEUR_INIT
136    BL MARCHE
137    ; Activer les deux moteurs droit et gauche
138
139    ;*****
140
141
142    MARCHE
143        BL MOTEUR_DROIT_ON
144        BL MOTEUR_GAUCHE_ON
145        BL MOTEUR_DROIT_AVANT
146        BL MOTEUR_GAUCHE_AVANT ; Activer le moteur et le Robot Avance
147        BL ALLUMER_LED1
148        BL ALLUMER_LED2 ; Allumer led1 ou led2
149        BL VERIF_BUMPER1 ; test bumpers 2
150        BL VERIF_BUMPER2 ; test bumpers 1
151        BL VERIF_SWITCH1 ; verifie switch1
152        BL VERIF_SWITCH2 ; verifie switch2
153
154        B MARCHE ; revenir à la marche
155
156
157    VERIF_BUMPER1
158        ldr r0,[r9] ; r0 la valeur du bumper 1
159        CMP r0,#0x00 ; test si bumper 1 appuyé
160        BEQ ARRETER ; arrête le robot
161
162        BX LR ; sinon on revient à la ligne suivant du programme // on execute VERIF_BUMPER2
163
164    VERIF_BUMPER2
165        ldr r0,[r10] ; r0 la valeur du bumper 2
166        CMP r0,#0x00 ; test si bumper 2 appuyé lorsque r0 égale 0x00
167        BEQ ARRETER ; arrêter le robot
168
169        BX LR ; sinon on revient à la ligne suivant du programme
170
```

```
171    VERIF_SWITCH1
172        ldr r6,[r11] ; r6 la valeur du switch1
173        CMP r6,#0x00 ; test si switch1 appuyer
174        BEQ TOURNER_DROIT ; le robot tourne à droite
175        BX LR ; le robot avance
176
177    VERIF_SWITCH2
178        ldr r6,[r12] ; r6 prend la valeur du switch2
179        CMP r6,#0x00 ; test si switch2 appuyer
180        BEQ TOURNER_GAUCHE ; si oui , executer le robot tourne à gauche
181        BX LR ;le robot avance
182
183
184    ALLUMER_LED1
185        str r2,[r8] ; éteindre une led1&2
186        str r3,[r7] ; allumer led1
187        BX LR
188
189
190
191
192    ALLUMER_LED2
193        str r2,[r7] ; éteindre une led1&2
194        str r4,[r8] ; allumer led2
195        BX LR
196
197
198
199
200    ARRETER
201        BL MOTEUR_GAUCHE_OFF ; désactiver le robot
202        BL MOTEUR_DROIT_OFF ; pendant une période
203        BL WAIT_ARRET ; puis réactiver le robot
204        B MARCHE
205
206
207    TOURNER_DROIT
208        BL ALLUMER_LED1 ; allumer led1
209        BL MOTEUR_DROIT_ARRIERE ; tourner à droit pendant une période
210        BL WAIT_TOURNER ; puis avancer
211        B MARCHE
212
213
```

```
213
214    WAIT_ARRET
215        ldr r1, = DUREE_ARRET ; r1 prend la valeur de durée tourner
216    wait1    subs r1, #1
217        bne wait1 ;branchement pour tester à nouveau wait1
218        BX LR
219
220
221    WAIT_TOURNER ; durée pour tourner le robot soit à droite , soit au gauche
222        ldr r1, = DUREE_TOURNER ; r1 prend la valeur de durée tourner
223    wait2    subs r1, #1
224        bne wait2 ; branchement pour tester à nouveau wait2
225        BX LR
226
227
228    TOURNER_GAUCHE
229        BL ALLUMER_LED2
230        BL MOTEUR_GAUCHE_ARRIERE
231        BL WAIT_TOURNER ; tourne à gauche pendant une periode
232        B MARCHE ; revenir à Marche
233
234
235
236
237        NOP
238        END
```