PROJET EVABOT

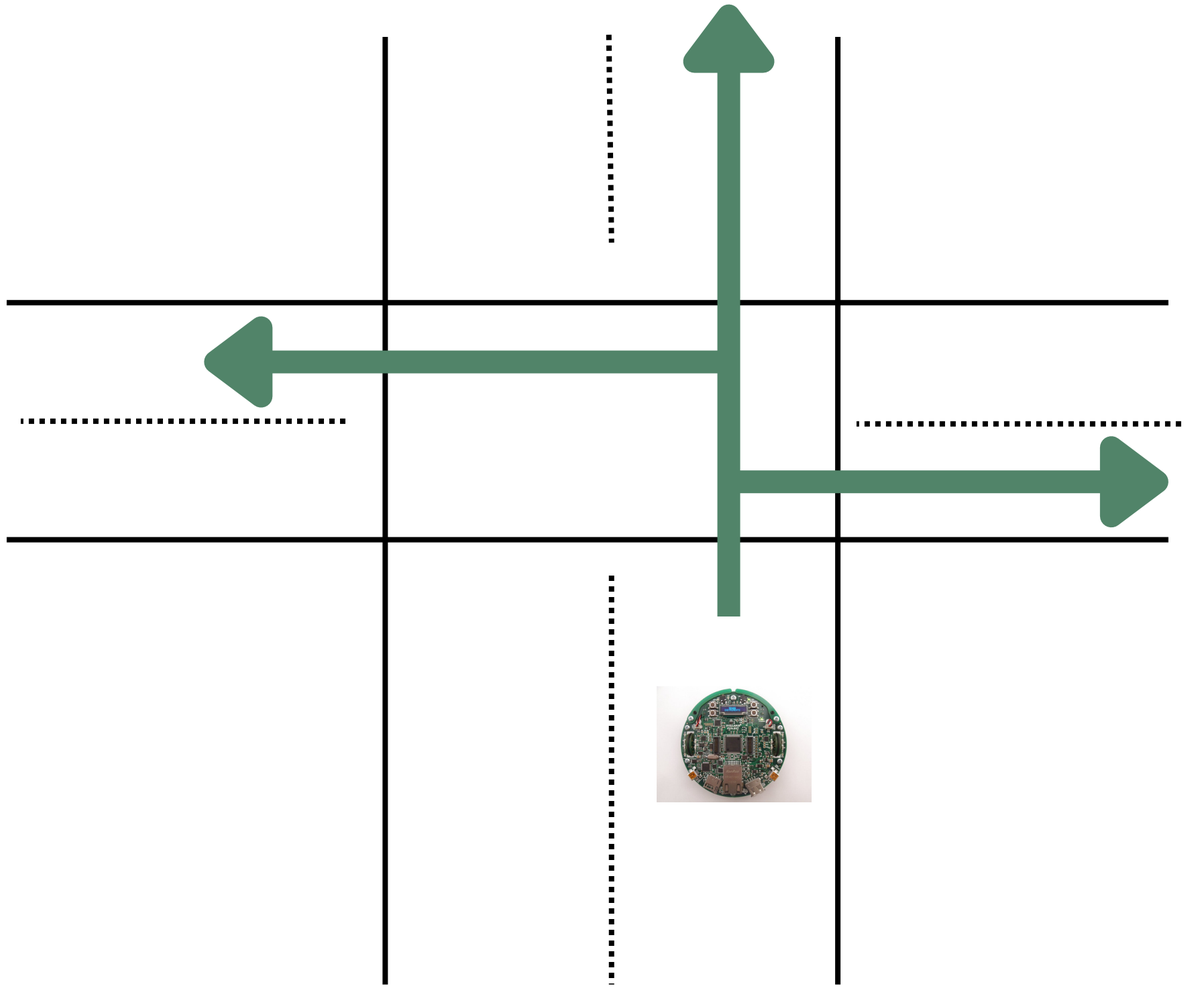
AUTOBOT

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)
;vvvvvvvvvvvvvvvvvvvvvvvvvvFin 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)
;vvvvvvvvvvvvvvvvvvvvvvvvvvFin configuration Bumper
```

```
;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^CONFIGURATION Bumper 1 & 2

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

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

ldr r9, = GPIO_PORTE_BASE + (BROCHE0<<2)
ldr r10, = GPIO_PORTE_BASE + (BROCHE1<<2)
;vvvvvvvvvvvvvvvvvvvvvvvvvvFin 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 droit
        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 droit 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)

```
        AREA    |.text|, CODE, READONLY

SYSCTL_PERIPH_GPIO  EQU     0x400FE108

GPIO_PORTF_BASE     EQU     0x40025000

GPIO_PORTE_BASE     EQU     0x40024000

GPIO_PORTD_BASE     EQU     0x40007000

;------------------------------------

GPIO_O_DIR          EQU     0x00000400

GPIO_O_DR2R         EQU     0x00000500

GPIO_O_DEN          EQU     0x0000051C

GPIO_I_PUR          EQU     0x00000510

;------------------------------------

BROCHE4         EQU     0x10        ; led1 sur broche 4
BROCHE5         EQU     0x20        ; led2 sur broche 5
BROCHE0         EQU     0x01        ; bumper 1
BROCHE1         EQU     0x02        ; bumper 2
BROCHE6         EQU     0x40        ; bouton poussoir 1
BROCHE7         EQU     0x80        ; bouton poussoir 2

;------------------------------------

; blinking frequency
DUREE_TOURNER   EQU     0x01383F7F
DUREE_ARRET     EQU     0x4c4b40
```

```
        ENTRY
        EXPORT  __main

        ;; The IMPORT command specifies that a symbol is defined in a shared object at runtime.
        IMPORT  MOTEUR_INIT                 ; initialise les moteurs (configure les pwms + GPIO)

        IMPORT  MOTEUR_DROIT_ON             ; activer le moteur droit
        IMPORT  MOTEUR_DROIT_OFF            ; désactiver le moteur droit
        IMPORT  MOTEUR_DROIT_AVANT          ; moteur droit tourne vers l'avant
        IMPORT  MOTEUR_DROIT_ARRIERE        ; moteur droit tourne vers l'arrière

        IMPORT  MOTEUR_GAUCHE_ON            ; activer le moteur gauche
        IMPORT  MOTEUR_GAUCHE_OFF           ; désactiver le moteur gauche
        IMPORT  MOTEUR_GAUCHE_AVANT         ; moteur gauche tourne vers l'avant
        IMPORT  MOTEUR_GAUCHE_ARRIERE       ; moteur gauche tourne vers l'arrière


__main
        ldr r6, = SYSCTL_PERIPH_GPIO
        mov r0, #0x00000038
        str r0, [r6]


        nop
        nop
        nop
```

```
        ;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^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)
        ;vvvvvvvvvvvvvvvvvvvvvvvvFin 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)
        ;vvvvvvvvvvvvvvvvvvvvvvvvFin configuration Bumper
```

```
        ;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^CONFIGURATION Bumper 1 & 2

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

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

        ldr r9, = GPIO_PORTE_BASE + (BROCHE0<<2)
        ldr r10, = GPIO_PORTE_BASE + (BROCHE1<<2)
        ;vvvvvvvvvvvvvvvvvvvvvvvvFin configuration Bumper
```

```
        ;; BL Branchement vers un lien (sous programme)
        str r9, [r7]
        str r4, [r8]
        ; 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


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
```

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


ALLUMER_LED1
        str r2,[r8]         ;; éteindre une led1&2
        str r3,[r7]         ;;  allumer led1
        BX LR



ALLUMER_LED2
        str r2,[r7]           ;; éteindre une led1&2
        str r4,[r8]           ;; allumer led2
        BX LR



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


TOURNER_DROIT
        BL ALLUMER_LED1             ;; allumer led1
        BL MOTEUR_DROIT_ARRIERE
        BL WAIT_TOURNER               ;; tourner à droit 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




        NOP
        END
```