



Livrable 2 - Robot Kinocto

Design III

présenté à

M. Dominique Grenier, M. Luc Lamontagne et M. Abdelhakim Bendada

<i>matricule</i>	<i>nom</i>
910 058 073	Émile Arsenault
908 190 985	Philippe Bourdages
910 098 468	Pierre-Luc Buhler
998 107 355	Diane Fournier
908 159 170	Imane Mouhtij
908 318 388	Olivier Sylvain
910 055 897	Daniel Thibodeau
910 097 879	Francis Valois

Université Laval
7 mars 2013

Chapitre 1

Introduction

Ce rapport présente la première phase de développement entourant le projet Kinocto. Les entrées et sorties (tâches) associées à la conception du robot sont mises en perspective dans le diagramme de contexte (voir section ??). Les exigences et les fonctionnalités issues de la présentation du projet par le client sont liées à travers le diagramme des propriétés fonctionnelles, présenté à la section 9.

Chapitre 2

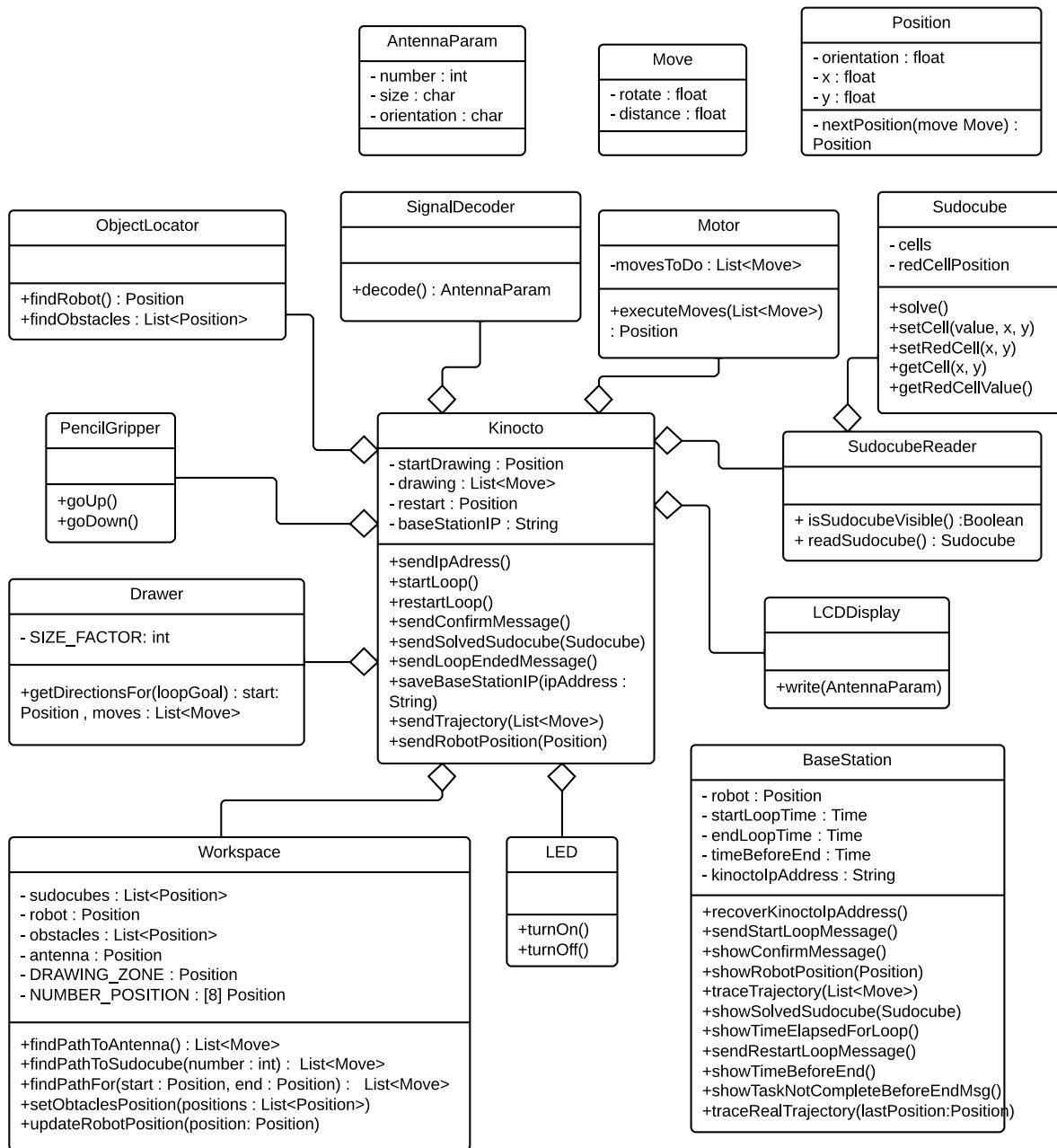
Diagramme des fonctionnalités

Chapitre 3

Diagramme physique

Chapitre 4

Diagrammes de classes



Chapitre 5

Diagrammes de séquences

Ce chapitre présente les différentes figures associées au diagramme des séquences. Ce diagramme est séparé selon cinq portions relatives aux fonctions particulières du robot. La figure 5.1 présente les liens entre l'utilisateur et la kinocto. La figure 5.2 présente les liens entre la kinocto et son environnement afin de déterminer sa position. La figure 5.3 présente les liens entre la kinocto et la station de base pour la transmission et l'affichage de la trajectoire optimale. La figure 5.4 présente les liens entre la kinocto et la table de jeu dans l'optique du déplacement de celle-ci à travers les obstacles vers la zone de lecture ainsi que les liens avec la résolution du sudocube. La figure 5.5 présente les liens entre la kinocto et ses différents périphériques lors de la production du dessin et de la confirmation de la résolution de la tâche.

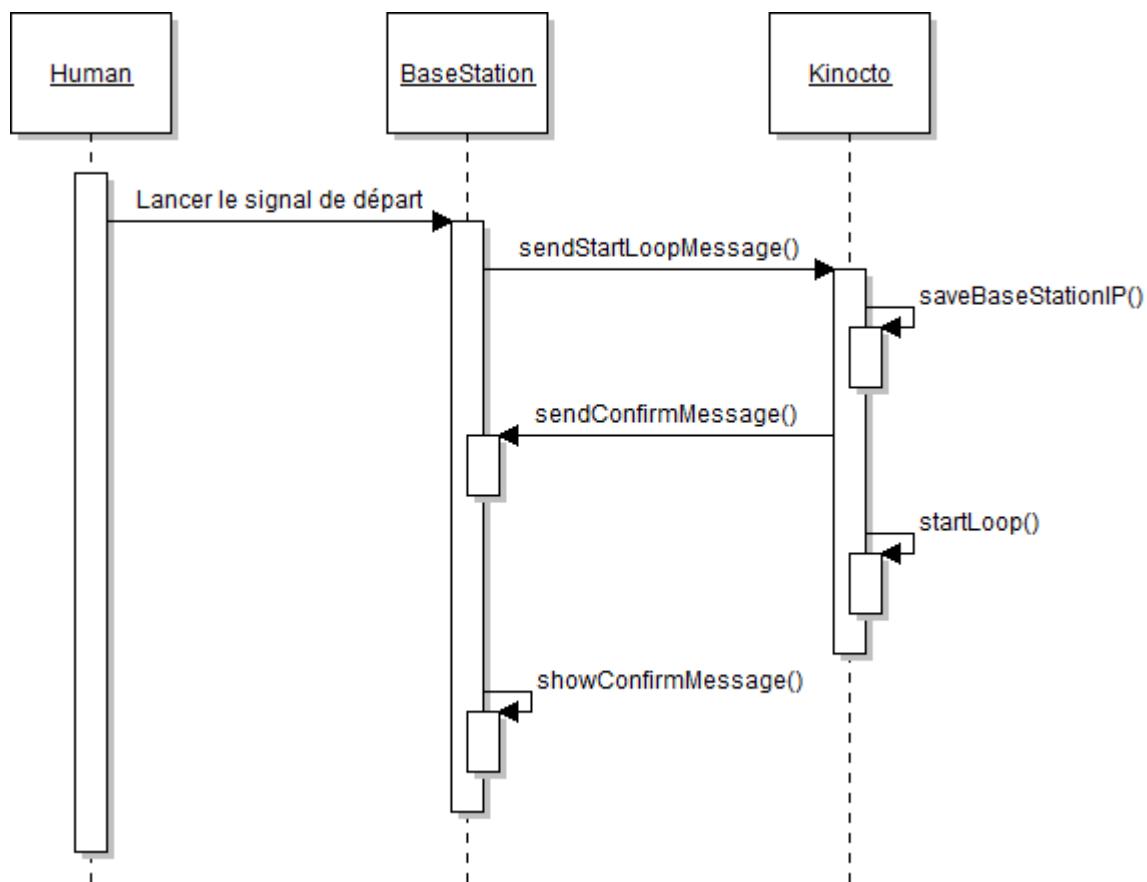


Figure 5.1 – Diagramme des séquences présentant les liens entre l'utilisateur et la kinocto

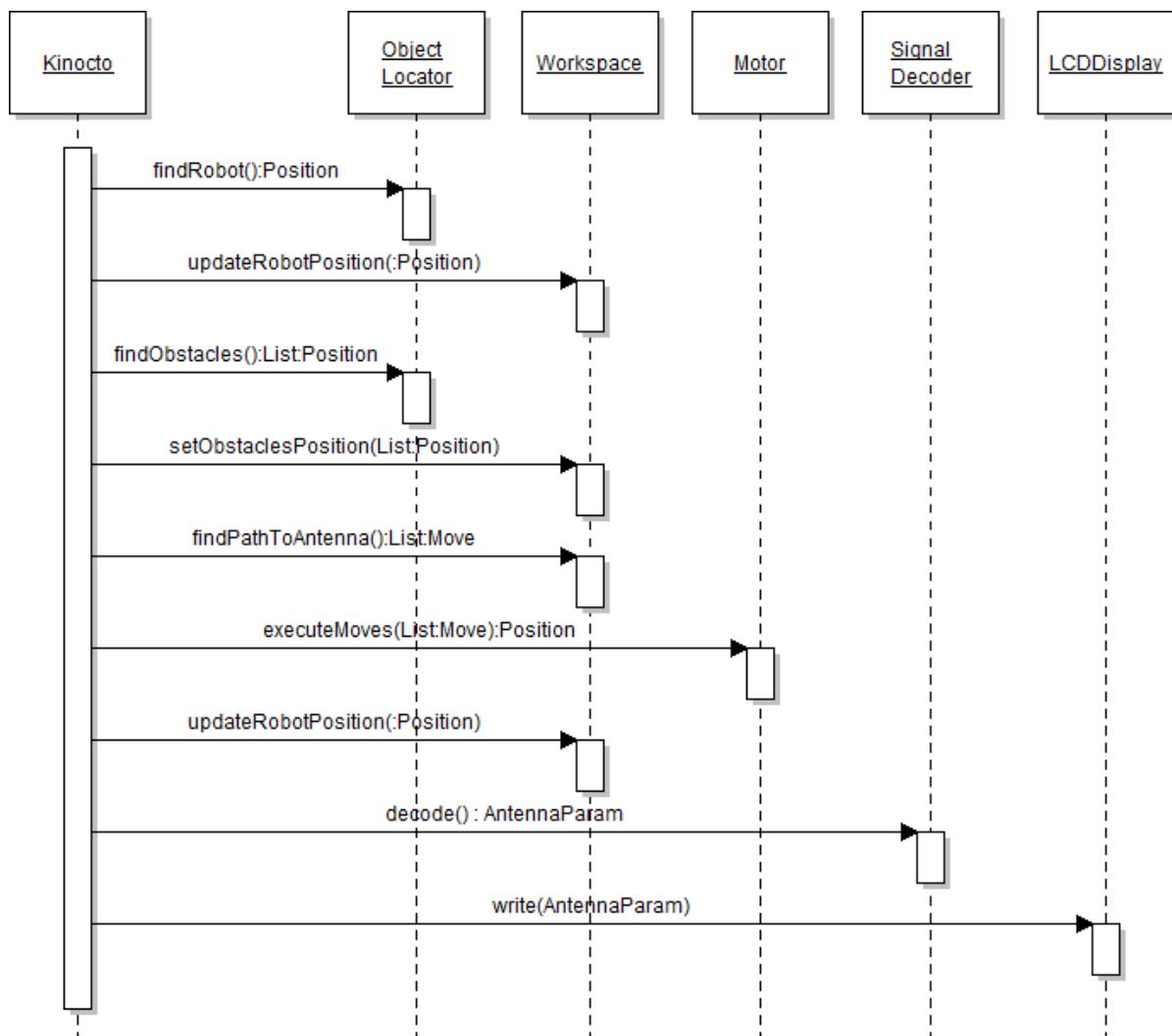


Figure 5.2 – Diagramme des séquences présentant les liens entre la *kinocto* et son environnement afin de déterminer sa position

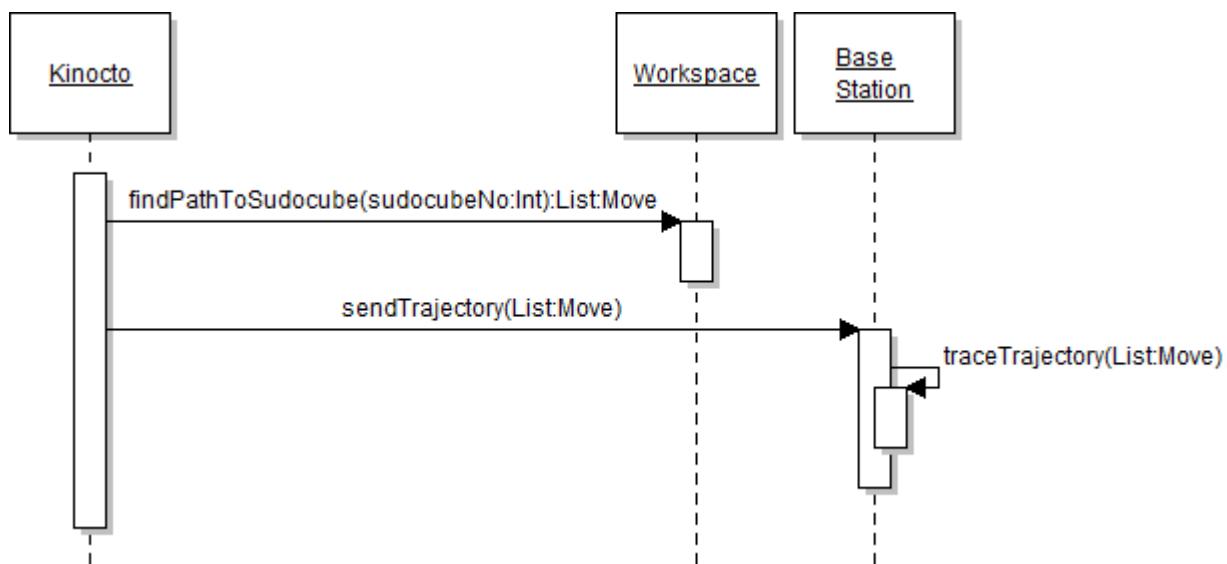


Figure 5.3 – Diagramme des séquences présentant les liens entre la kinocto et la station de base pour la transmission et l'affichage de la trajectoire optimale

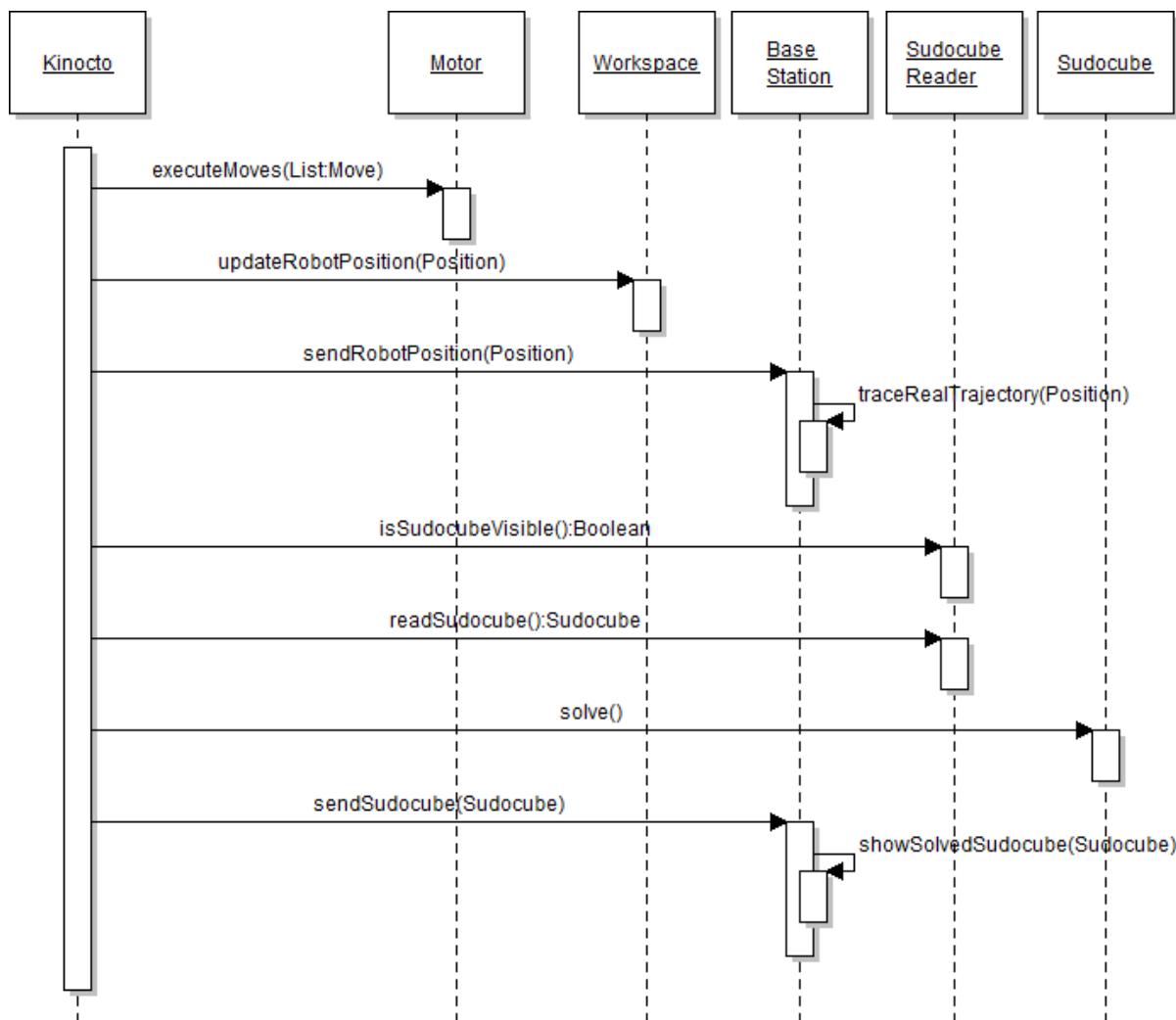


Figure 5.4 – Diagramme des séquences présentant les liens entre la kinocto et la table de jeu dans l'optique du déplacement de celle-ci à travers les obstacles vers la zone de lecture ainsi que les liens avec la résolution du sudocube

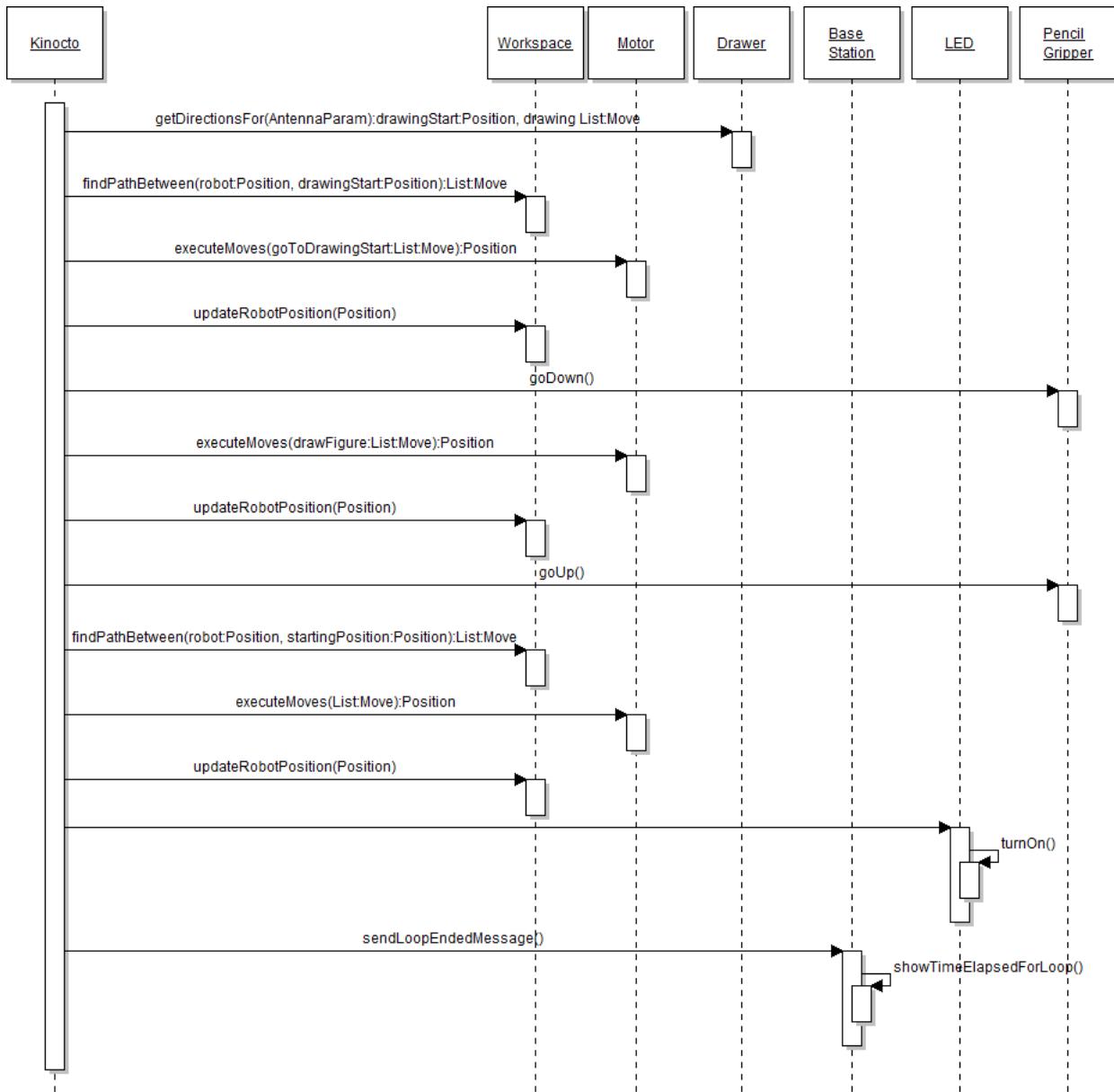


Figure 5.5 – Diagramme des séquences présentant les liens entre la kinoclo et ses différents périphériques lors de la production du dessin et de la confirmation de la résolution de la tâche

Chapitre 6

Registre de risques

Tableau 6.1 – Registre de risques partie 1

Risque	Type de risque	Niveau de priorité (1-faible, 5-élevé)	Conséquences de l'occurrence du risque	Coût en performance associé au risque	Prob. d'occurrence (%)	Coût estimé du risque (\$)	Plan de réduction du risque	Responsable du risque
1	Bris de la batterie Li-Po suite à une mauvaise utilisation	5	Plus d'autonomie énergétique pour le robot, opération impossible	L'ensemble du système sera inopérable	15	60	Formation des utilisateurs à l'endroit de l'utilisation. Apposition d'un capteur de tension. Dispositifs de protection (fusibles)	É. Arsenault
2	Bris d'un ou des systèmes d'alimentation	5	Les systèmes auxiliaires, le Mac mini ou les moteurs ne seront pas alimentés. Le système ne sera pas fonctionnel	Une partie ou l'ensemble du système sera inopérable	20	25	Utilisateur de connecteurs protégés (d'un seul sens possible). Dispositifs de protection (fusibles), surdimensionnement des composantes d'alimentation	D. Thibodeau
3	Bris du crayon lors du dessin	4	Le dessin ne pourra être complété	La portion dessin sera partiellement complète	20	2	Tests rigoureux et optimisation du choix de crayon avant la compétition	É. Arsenault
4	Bris du système de préhension	5	Le dessin ne pourra être complété et selon le moment du bris, la trajectoire du robot peut être affectée	La portion dessin sera partiellement complète et la trajectoire sera déviée	15	10	Tests rigoureux et essais multiples pour vérifier la stabilité en température lors du fonctionnement	É. Arsenault
5	Problèmes d'alimentation électrique (surtension ou baisse de tension, surchauffe ou ondulation importante)	4	Une surtension importante pourrait être destructrice pour le Mac mini ou le microcontrôleur, une surchauffe causerait un bris des systèmes d'alimentation et une ondulation importante pourrait rendre le système instable	Conséquences des risques 1 et 2 en plus de bris du Mac mini ou du microcontrôleur	10	100	Tests rigoureux et essais multiples pour vérifier la stabilité en température, en tension et en courant lors du fonctionnement	D. Thibodeau

Tableau 6.2 – Registre de risques partie 2

Risque	Type de risque	Niveau de priorité (1-faible, 5-élevé)	Conséquences de l'occurrence du risque	Coût en performance associé au risque	Prob. d'occurrence (%)	Coût estimé du risque (\$)	Plan de réduction du risque	Responsable du risque
6	Problèmes de réception du signal d'antenne	4	Si la réception est erronée, le robot peut exécuter sa séquence, mais l'exécution ne sera pas conforme au message. Si la réception est impossible, le système ne s'amorcera pas.	Accomplissement de la mauvaise tâche ou système non fonctionnel	5	15	Tests répétés pour un taux de succès de 100% lors de la réception et du décodage.	D.Fournier
7	Problème d'identification du robot et de l'environnement (vision) par la Kinect	5	Correction de la trajectoire erronée, risque de rencontrer les obstacles, mauvaise trajectoire calculée	La trajectoire réelle ne sera pas optimale et le robot risque de rencontrer des obstacles	10		Tests rigoureux et taux de succès de l'identification proche de 100%	I. Mouhtij
8	Problème de détection des chiffres et de représentation du sudo cube	5	Erreur dans la résolution du sudo cube (mauvais chiffres à la base)	Le chiffre dessiné ne sera pas le bon	5		Tests répétés pour un taux de succès de près de 100% lors de la détection des chiffres	P. Bourdages
9	Problème de résolution du sudo cube	5	Erreur dans la résolution du sudo cube, emballlement de l'algorithme et arrêt de la tâche	Le chiffre dessiné ne sera pas le bon, le robot peut arrêter sa séquence d'opération avant le moment prévu	15		Tests répétés pour un taux de succès de près de 100% lors de la résolution du cube	O. Sylvain
10	Problème de communication entre le Mac mini et la station de base	5	Les informations requises ne pourront être transmises correctement, on perd l'information sur le comportement du robot ainsi que sa position. Le robot ne pourra pas se localiser initialement et en temps réel.	Le robot ne remplira pas les exigences d'affichage sur la station de base, le robot ne recevra aucune position de la Kinect	5		Tests répétés pour un taux de succès de près de 100% lors de la transmission et de la réception de l'information en temps réel entre le Mac mini et la station de base	P. Bourdages

Tableau 6.3 – Registre de risques partie 3

Risque	Type de risque	Niveau de priorité (1-faible, 5-élevé)	Conséquences de l'occurrence du risque	Coût en performance associé au risque	Prob. d'occurrence (%)	Coût estimé du risque (\$)	Plan de réduction du risque	Responsable du risque
11	Problème de communication entre le Mac mini et le microcontrôleur	5	L'ensemble des auxiliaires ne fonctionnera pas correctement. Le système sera non fonctionnel	Robot incapable de se déplacer selon la trajectoire prévue et d'effectuer la tâche	5		Tests répétés pour un taux de succès de près de 100% lors de la transmission et de la réception de l'information entre le Mac mini et le microcontrôleur	D. Fournier
12	Asservissement des moteurs déficient	3	Un asservissement instable causera des dépassemens de positions et un temps de réponse plus long du moteur aux consignes. Dans le pire des cas, le robot deviendrait incontrôlable.	Robot plus lent ou incapable de se déplacer, hausse du risque de toucher les obstacles, système non fonctionnel	10		Tests rigoureux dans le plus de conditions possible. Maximiser la robustesse en réduisant le temps de réponse.	P. Buhler
13	Contact avec un ou des obstacles	3	Bris du système et déviation de trajectoire possible	Un robot qui entre en contact avec les obstacles ne remplit pas les exigences du projet	20		Tests rigoureux sur les déplacements et marge de sécurité importante pour le contournement.	O. Sylvain
14	Bris d'une portion ou de la totalité du microcontrôleur	4	Le bris d'une portion du microcontrôleur empêche l'exécution de la tâche dans son ensemble et peut occasionner des bris dans les systèmes reliés	Un robot qui ne peut pas se déplacer correctement, qui n'allume pas la DEL ou qui n'active pas le préhenseur	5	100	Isolation des entrées et sorties avec des dispositifs de protection (diode), limiteur de courant	D. Thibodeau
15	Bris du pont en H	5	Les moteurs ne pourront être commandés correctement	Le robot ne peut pas se déplacer, le système n'est pas fonctionnel	5	130	Utilisation d'un régulateur de type PID afin d'éviter les appels brusques de courants, positionnement du pont de manière à limiter son exposition aux accrochages	F. Valois

Tableau 6.4 – Registre de risques partie 4

Risque	Type de risque	Niveau de priorité (1-faible, 5-élevé)	Conséquences de l'occurrence du risque	Coût en performance associé au risque	Prob. d'occurrence (%)	Coût estimé du risque (\$)	Plan de réduction du risque	Responsable du risque
16	Bris du Mac mini	5	L'ensemble des auxiliaires ne fonctionnera pas correctement. Le système sera non fonctionnel	Robot incapable de se déplacer selon la trajectoire prévue et d'effectuer la tâche	5	600	Apposition de protections électriques (fusibles et interrupteurs) sur l'étage d'alimentation du Mac mini. Fixation robuste du Mac mini sur le robot. Protection du port USB du Mac mini en n'utilisant pas le fil d'alimentation.	F. Valois
17	Bris du système d'exploitation du Mac mini	4	La portion logicielle du robot et le traitement seront absents. Le système ne sera pas fonctionnel	Robot incapable d'accomplir un traitement de tâches	30		Clonage d'une version fonctionnelle et stable du système d'exploitation	P. Buhler
18	Caméra web désaxée	3	Les prises de données du sudo cube seront affectées	L'acquisition des données du sudo cube pourrait être non fonctionnelle et causer une erreur dans la résolution du cube	10		Tests rigoureux d'asservissement de la caméra et de retour à l'axe désiré suivant une modification externe.	P. Buhler
19	Bris de la caméra web	4	Le bris de la caméra empêche la vision des cubes	Si la caméra ne peut voir le sudo cube, on ne peut trouver le chiffre dans la case rouge et effectuer le bon dessin	5	80	Storage adéquat de la caméra, protection d'alimentation (fusible), limiter les chocs contre les obstacles	D. Thibodeau

Chapitre 7

Plan de tests avec matrice de vérification des exigences intégrées

Chapitre 8

Plan d'intégration

Chapitre 9

Description des propriétés fonctionnelles

Pour simplifier la lecture du tableau de la description des propriétés fonctionnelles, celui-ci a été séparé en trois pages selon trois différentes sections : vision et traitement numérique (présenté dans le tableau 9.1), communication et déplacement (présenté dans le tableau 9.2) ainsi qu'alimentation et affichage (présenté dans le tableau 9.3).

Tableau 9.1 – Description des propriétés fonctionnelles : section "Vision et Traitement Numérique"

Tableau 9.2 – Description des propriétés fonctionnelles : section "Communication et Déplacement"

Exigences du client	Fonctionnalités							Résolu- tion (Degrés)	Vitesse (m/s)
	Communication						Déplacement		
	Recevoir le signal d'antenne	Communiquer entre le robot et la station de base	Communiquer entre le Mac mini et le microcontrôleur	Commander les moteurs	Transmettre les images de la caméra vers le Mac	Contrôler la position de la caméra	Commander le préhenseur du crayon		
Être autonome pendant un minimum de 10 minutes	4							3	1
Se déplacer selon la trajectoire optimale	3		3	4	2			5	5
Effectuer une séquence complète en moins de 10 minutes	5		3	4				5	5
Alimenter le Mac mini avec une tension de 22V à 30V et une ondulation de tension inférieure à 200 mV									
Résoudre sudocube					3	1			
Dessiner le chiffre selon le signal d'antenne dans une zone prédéfinie (jaune) avec une précision de $\pm 1\text{cm}$			5	3				5	5
éviter les obstacles			5	3	4			5	
Analyser le bon cube selon le signal d'antenne	5								
Utiliser la communication sans fil									
Concevoir un système de préhension pour le crayon								5	
Afficher position réelle				5					
Afficher la trajectoire optimale	3	5							
Afficher le cube résolu la base		5			3				
Allumer une DEL lorsque tâche terminée			5						
Afficher message de fin		5		5					
Afficher message de départ									
Afficher trajectoire réelle avec un délai maximum de 15s									
Afficher les informations sur le robot									
Respecter un budget de 250\$	3		1	5			2		

Tableau 9.3 – Description des propriétés fonctionnelles : section "Alimentation et affichage"

	Fonctionnalités										
	Alimentation				Affichage						
	Utiliser une pile rechargeable	Alimenter les moteurs	Alimenter le Mac	Alimenter les différents périphériques	Afficher message d'initiation de la tâche	Afficher le cube résolu	Allumer la DEL lorsque tâche complétée	Afficher la trajectoire optimale	Afficher la position réelle	Afficher message de fin	Afficher sur le LCD
Exigences du client		Puissance (W)		Ondulation de tension (V)					Temps d'actualisation (s)		
Être autonome pendant un minimum de 10 minutes	5	3	3	3							
Se déplacer selon la trajectoire optimale	5	5	5	3							
Effectuer une séquence complète en moins de 10 minutes											
Alimenter le Mac mini avec une tension de 22V à 30V et une ondulation de tension inférieure à 200 mV			5	5							
Résoudre sudocube	5		5	5							
Dessiner le chiffre selon le signal d'antenne dans une zone prédéfinie (jaune) avec une précision de $\pm 1\text{cm}$	3	3	3	3							
éviter les obstacles	3	3	3	3							
Analyser le bon cube selon le signal d'antenne	5		1	3	3	3	3				
Utiliser la communication sans fil											
Concevoir un système de préhension pour le crayon											
Afficher position réelle			1	3			3				5
Afficher la trajectoire optimale	3		1	3	3	3	3				5
Afficher le cube résolu sur la base		5	1		3	3	3	5			
Allumer une DEL lorsque tâche terminée			1				2		5		
Afficher message de fin											
Afficher message de départ				3	5						
Afficher trajectoire réelle avec un délai maximum de 15s				3					5		
Afficher les informations sur le robot				3							5
Respecter un budget de 250\$	5	2	3	5							

Chapitre 10

Développement logiciel

Chapitre 11

Test unitaires

Chapitre 12

Avancement de la conception et de la construction du système

12.1 Alimentation des périphériques 5V

L'alimentation employée pour les périphériques est une alimentation de type buck qui convertit la tension de la batterie (11.1V) vers une tension usuelle de 5V. Cette alimentation utilise un hacheur de tension avec une fréquence autour de 50kHz. Cette fréquence procure une marge de sécurité par rapport à la fréquence d'antenne et évite l'ajout d'un bruit excessif. Cette alimentation a été réalisée avec succès, les plans sont présentés à la figure ???. Une photo de ladite alimentation est présentée à la figure ???.

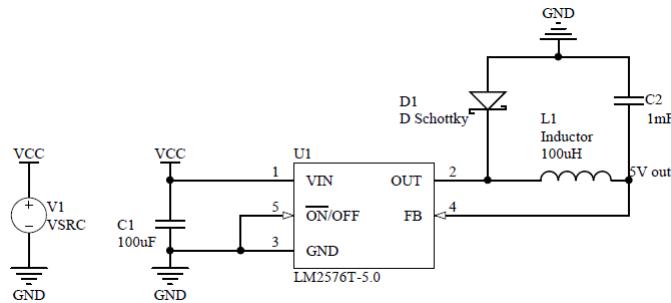


Figure 12.1 – Figure présentant les plans de l'alimentation 5V pour les périphériques

12.2 Alimentation du Mac mini

L'alimentation du Mac mini a été réalisé au moyen d'un circuit de type Boost réglable, acheté déjà monté. Les plans ne sont pas disponibles, mais une photo du dispositif l'est à la figure ???.

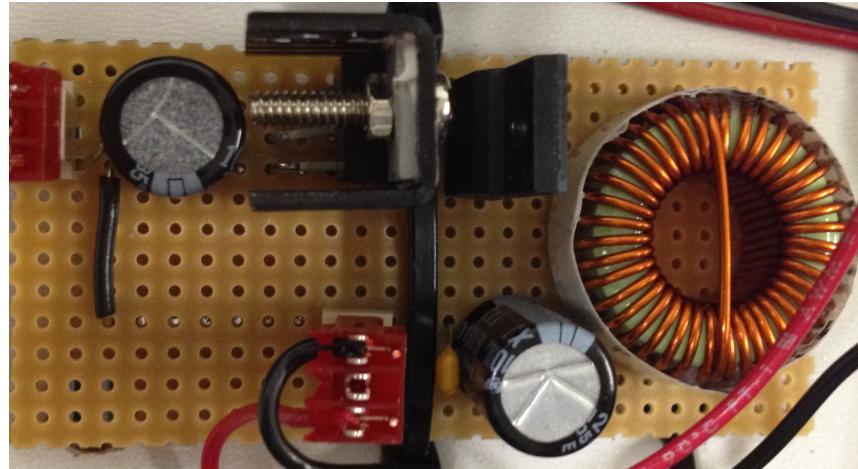


Figure 12.2 – Figure présentant une photo de l'alimentation 5V pour les périphériques

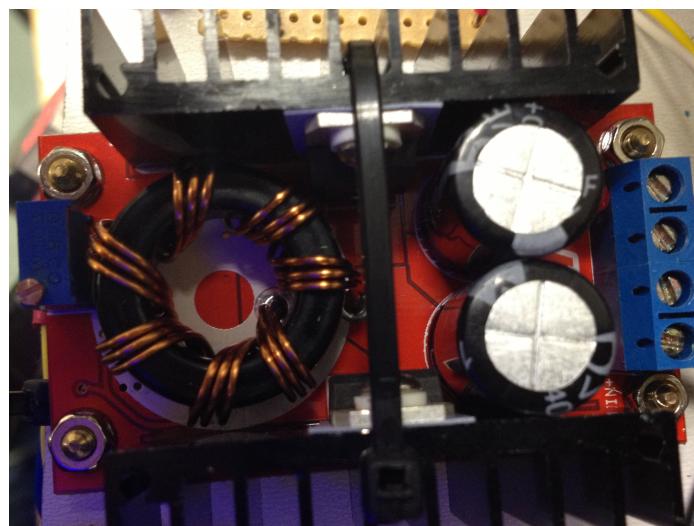


Figure 12.3 – Figure présentant une photo de l'alimentation 24V pour les périphériques

Annexe A

Annexes

A.1 Code test pour affichage sur LCD

A.1.1 ecran.h

```
1 #ifndef ECRAN_H_
2 #define ECRAN_H_
3
4 #include "inc/hw_ints.h"
5 #include "inc/hw_memmap.h"
6 #include "inc/hw_types.h"
7 #include "driverlib/debug.h"
8 #include "driverlib/gpio.h"
9 #include "driverlib/interrupt.h"
10 #include "driverlib/rom.h"
11 #include "driverlib/timer.h"
12 #include "inc/lm3s9b92.h"
13
14 // definition de trucs qui vont etre pratiques
15
16 //#define STCTRL (*((volatile unsigned long *)0xE000E010))
17 //#define STRELOAD (*((volatile unsigned long *)0xE000E014))
18 //#define attend(t) SysCtlDelay((SysCtlClock() / 3) / (1000 / t))
19
20
21 #define ECRAN_DATA GPIO_PORTE_DATA_R
22 #define ECRAN_CTRL GPIO_PORTA_DATA_R
23 // volatile unsigned long ulLoop;
24
25 // bit de data du LCD
26 #define ECRAN_D0 0x01 //PE0
27 #define ECRAN_D1 0x02 //PE1
28 #define ECRAN_D2 0x04 //PE2
29 #define ECRAN_D3 0x08 //PE3
30 #define ECRAN_D4 0x10 //PE4
31 #define ECRAN_D5 0x20 //PE5
32 #define ECRAN_D6 0x40 //PE6
33 #define ECRAN_D7 0x80 //PE7
34
35 // bit de control du LCD
36 #define ECRAN_RS 0x01 //PA0 reset
37 #define ECRAN_RW 0x02 //PA1 read/write
38 #define ECRAN_EN 0x04 //PA2 enable
39 #define ECRAN_BF 0x08 //PA3 "busy flag", indique que l'écran est occupé
40
41 // volatile unsigned long ulLoop;
42 void ecranClear(void);
43 void ecranInit(void);
44 void ecranWriteChar(char caractere);
45 void ecranWriteLine(char * line, unsigned short size);
```

```

46 void ecranSetPosCursor( short pos);
47 void ecranAttend( void );
48 void ecranControl( unsigned long input );
49
50 #endif /*ECRAN_H*/

```

A.1.2 ecran.c

```

1 #include "ecran.h"
2
3
4 /*
5 * Cette fonction permet de s'assurer que l'écran n'est pas occupé avant d'écrire
6 */
7 void ecranAttend( void )
8 {
9     ECRAN_CTRL &= ~(ECRAN_RS); // RS = 0
10    ECRAN_CTRL |= ECRAN_RW; // RW = 1
11    volatile unsigned long busyflag = ECRAN_CTRL & ECRAN_BF;
12    while (busyflag != 0) // on regarde le busyflag pour s'assurer que l'écran est pas occupé
13    {
14        busyflag = GPIO_PORTD_DATA_R & ECRAN_BF;
15    }
16    return;
17 }
18
19 /*
20 * Cette fonction exécute la routine d'initialisation de l'écran dans le mode qu'on veut
21 */
22 void ecranInit( void )
23 {
24     ecranControl(ECRAN_D5 | ECRAN_D4 | ECRAN_D3 | ECRAN_D2);
25     ecranControl(ECRAN_D3 | ECRAN_D2 | ECRAN_D1);
26 }
27
28 /*
29 * Cette fonction permet de faire des commandes tel que l'initialisation , changement de positions et
30 * trucs du genre, tout ce qui est pas l'écriture de caractères sur l'écran
31 */
32 void ecranControl( unsigned long input )
33 {
34     ecranAttend();
35     volatile unsigned long ulLoop;
36     ECRAN_DATA = input; // on met notre entrée sur le port de données
37     ECRAN_CTRL &= ~(ECRAN_RS | ECRAN_RW); // reset et read/write à zero pour pas qu'il re-écrive par erreur
38     ECRAN_CTRL ^= ECRAN_EN;
39     ulLoop = SYSTIMER_RCGC2_R;
40         for (ulLoop = 0; ulLoop < 200; ulLoop++)
41         {
42             // on met un délai pour que le LCD puisse voir le enable
43         }
44     ECRAN_CTRL ^= ECRAN_EN;
45     ulLoop = SYSTIMER_RCGC2_R;
46         for (ulLoop = 0; ulLoop < 200; ulLoop++)
47         {
48             // on attend un peu pour que l'instruction s'exécute avant de changer la pin 7
49         }

```

```

50
51 ECRAN_DATA &= ~(ECRAN_D7); //on reset la pin 7 parce elle output aussi le busy flag et on telle pas veux etre pogner dans ecranAttend()
52 }
53
54
55
56 /*
57 * Cette fonction efface le contenu de l'écran
58 */
59 void ecranClear(void)
60 {
61     ecranControl(ECRAN_D0);
62     volatile unsigned long ulLoop;
63
64     for (ulLoop = 0; ulLoop < 2000; ulLoop++)
65     {
66         //loop pour etre sur que ton est bien clearer
67     }
68 }
69
70 /*
71 * Cette fonction change la position du curseur d'écriture de l'écran
72 */
73 void ecranSetPosCursor(short pos)
74 {
75     ecranControl(GPIO_PIN_7 | pos);
76 }
77
78 /*
79 * Cette fonction permet d'écrire un caractère sur l'écran
80 */
81 void ecranWriteChar(char caractere)
82 {
83     volatile unsigned long ulLoop;
84     ecranAttend();
85     ECRAN_DATA = caractere;
86     ECRAN_CTRL &= ~(ECRAN_RW); //RW = 0
87     ECRAN_CTRL |= ECRAN_RS; // rs = 1
88     ECRAN_CTRL ^= ECRAN_EN; //En = 1
89     ulLoop = SYSCTL_RCGC2_R;
90     for (ulLoop = 0; ulLoop < 200; ulLoop++)
91     {
92         //petit délai, juste pour être sur
93     }
94     ECRAN_CTRL ^= ECRAN_EN; //En = 0
95 }
96
97
98 /*
99 * Cette fonction permet d'écrire une chaîne de caractères sur l'écran
100 */
101 void ecranWriteLine(char * line, unsigned short size) //todo
102 {
103     unsigned short i=0;
104     for(i=0; i <size;i++)
105     {
106         ecranWriteChar(line[i]);
107     }
108 }
```