# Compte Rendu

# TP – Système Embarqué Temps Réel

Membres du groupe :

* TRAN Van Duc
* YIN Yiliang

## Introduction

On s’attache dans ce compte rendu à décrire et à analyser le travail de notre groupe dans deux TPs de Système Embarqué Temps Réel :

* TP1 : Programmation d’un robot NXT sous OSEK
* TP2 : POSIX programming on the EV3 robot

Ces TPs nous aident à comprendre le principe d’un système embarqué multitâche en utilisant la norme OSEK, la communication et l’ordonnancement des tâches avec l’aide du langage C.

## TP1 – Programmation d’un robot NXT sous OSEK

### But du TP

Ce TP joue sur la programmation d’un robot LEGO NXT qui est équipé avec des capteurs de l’infrarouge, de l’ultrason et du choc. L’idée est de programmer le robot de sorte à ce qu’il suive les spécifications suivantes :

* Le fonctionnement principal est de suivre une piste noire avec l’aide du capteur de l’infrarouge.
* Le robot se met à reculer lorsqu’il détecte un obstacle proche au capteur ultrason.
* Il avance lorsqu’il détecte un choc sur son capteur arrière.

Enfin, nous ferons une étude dynamique de durée moyenne d’exécution afin d’étudier l’ordonnançabilité du système, et de choisir les périodes d’exécution des tâches au plus juste par rapport à la puissance de calcul afin d’augmenter la réactivité du système. La conception de ce système est décrite par le diagramme ci-dessous :

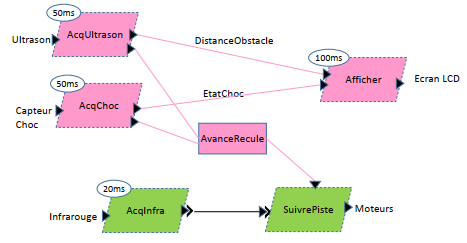


Figure : Conception du Système suivi piste

Pour le réaliser, on décide à deviser le système en deux partie :

* En vert est la stratégie principale, suivi piste.
* La partie en rose sert à détecter les obstacles autour le robot, réagir sur ces informations et les afficher sur l’écran LCD.

Il faut programmer en même temps, les tâches dans la source de code en langage c et la communication entre nos tâches dans le fichier .oil.

### Travail réalisé

Question 1 : Vérifier le fonctionnement de code.

Cette étape décrit comment compiler et lancer la source de code sur notre robot.

Question 2 : L’implémentation de l’étape 1 (partie en verte).

Ici, on constate deux tâches :

* Task\_AcqInfra qui
* Task\_SuivrePiste

D’abord, il faut déclarer ces deux tâches en norme OSEK.

TASK Task\_AcqInfra

{

AUTOSTART = FALSE;

PRIORITY = 1; /\* lowest priority \*/

ACTIVATION = 1;

SCHEDULE = FULL;

STACKSIZE = 512;

RESOURCE = sMonMDD;

};

TASK Task\_SuivrePiste

{

AUTOSTART = FALSE;

PRIORITY = 1; /\* lowest priority \*/

ACTIVATION = 1;

SCHEDULE = FULL;

STACKSIZE = 512;

RESOURCE = sMonMDD;

};

Pour la communication

Question 3 : L’implémentation de l’étape 2 (partie en rose).

Question 4

Question 5

Question 6

Question 7

## TP2 – POSIX programming on the EV3 robot

## Conclusion