P. Trau

LPSI3 Automatisme

TP3 Approfondissement : LIST et Grafcet

Aujourd'hui vous auriez pu travailler dans le langage LIST sur Siemens. Mais en fait vous utiliserez le Micro1, du moins un simulateur (fonctionnant sous Linux) disponible sur Moodle. Il est aussi utilisable sur Windows via DosBox. Cet automate est programmable dans un langage qui, comme le LIST, est en mode texte (langage mnémonique). Voyez la doc (disponible aussi sur Moodle). A priori les fonctions détaillées dans la doc en chapitre 5, jusqu'à 5.8, devraient vous suffire. Et bien sûr le chapitre 8. L'exemple tst-base.txt vous montrera l'usage de ces fonctions de base.

En télétravail, n'ayez pas peur de me poser des questions. Je peux vous permettre de partager votre écran, ou vous pouvez m'envoyer vos sources, je peux vous montrer, par partage de mon écran, certaines fonctionnalités du simulateur

Première partie : séquentiel (Grafcet) en List

Vous allez maintenant acquérir progressivement une méthode permettant de traiter tout Grafcet sur tout automate programmable (il lui suffit de posséder les fonctions de base du combinatoire (ET, OU, NON) et les bascules). Je vous ai dit en cours que de nombreuses entreprises internationales n'utilisent pas le Grafcet.

A - Perçage avec débourrage

On utilise

trois sorties :	montée M (sortie 200)
	descente D (sortie 201)
	rotation R (sortie 202)
quatre entrées :	haut h (entrée 0)
	milieu m (entrée 1)
	bas b (entrée 2)
	départ cycle dcy (entrée 3)

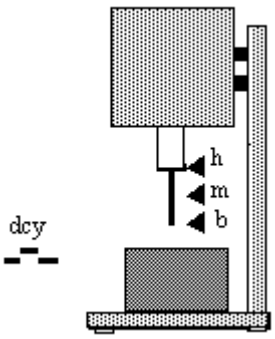
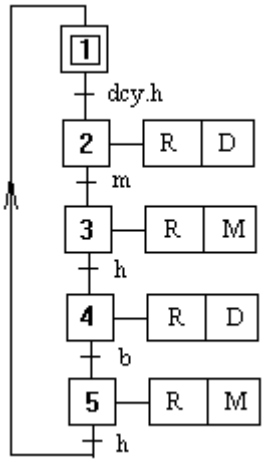


Schéma du dispositif
(échelle non respectée)



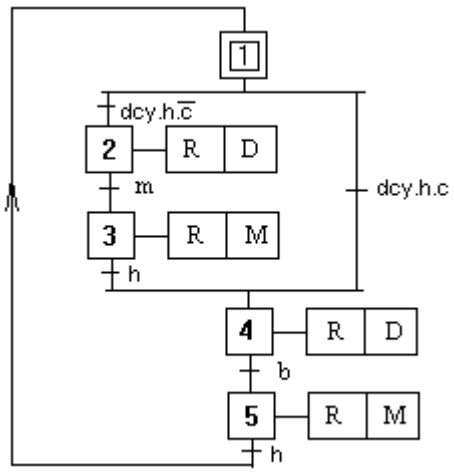
Cahier des charges : Au départ cycle (condition initiale broche en haut), descente de la broche jusqu'en m, remontée en h (pour débourrer, c.a.d sortir les copeaux), descente jusqu'en b puis remontée en h, pour attendre un nouveau dcy. Le foret doit tourner tout au long du cycle. Je vous impose le Grafcet à utiliser. Programmez ce Grafcet en respectant les consignes : chaque étape est représentée par une mémoire interne (400 à 405 par exemple) qui vaudra 1 lorsque l'étape est active, 0 sinon. Le programme se décomposera en 3 parties (3 réseaux distincts) : initialisation, évolution puis affectation des sorties, détaillées ci-dessous.

- A l'initialisation, on allume l'étape initiale et on éteint les autres mémoires. Utilisez l'entrée7 comme bouton Reset. (le Micro1 mettant tous les mémoires internes à 0 au début du programme, on pourrait également en utiliser une pour nous indiquer (lorsqu'elle vaut 0) que l'on est au premier cycle de programme, puis on la met définitivement à 1, mais je préfère un bouton reset.).

- L'évolution est traitée **transition par transition**. Une transition est franchie lorsque l'étape précédente est active et que sa réceptivité est vraie. On désactive alors l'étape précédente et on active la suivante. Vous utiliserez ici des « Set » et « Reset ». Vous ne traitez PAS les sorties ici !

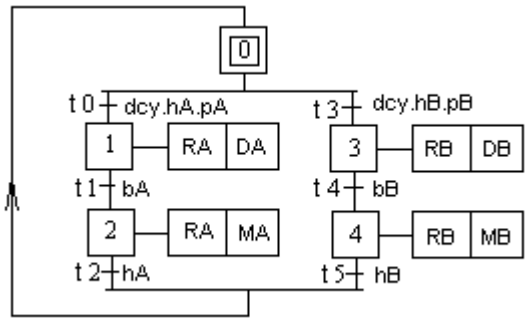
- Pour la clarté du programme, les sorties seront traitées obligatoirement après de -l'évolution : on modifie les sorties en fonction des étapes actives. Comme plusieurs étapes allument les mêmes sorties, vous utiliserez des « OU », une seule équation par sortie.

B - Rajoutons un interrupteur bistable sur l'entrée 4, commandé par l'opérateur, qui, s'il est appuyé, court-circuite le débourrage. Modifiez votre programme (il me semble qu'il suffit d'ajouter une transition). Le Grafcet est donné ci-contre (je suis gentil, non ?). Dans le programme, idem : il suffit d'ajouter une transition.



C - On dispose désormais de deux postes de perçage A et B (pour simplifier on ne prévoira plus de débourrage), avec un seul dcy mais deux capteurs de présence. Au dcy, on démarre les perçages pour les postes prêts (c'est à dire en position haute avec présence de pièce). Le Grafcet est donné ci-dessous. La méthode de programmation utilisée jusqu'ici ne respecterait pas les règles du Grafcet. Une bonne solution permettant de traiter ce cas (et d'ailleurs tous les cas) est de prévoir également une mémoire pour toutes les transitions (ici t0 à t5). On décompose la phase d'évolution en trois :

- calcul de toutes les transitions (définir si elles sont franchissables ou non) sans faire évoluer le Grafcet (ceci permet de respecter la règle 4 du Grafcet).
- désactivation de toutes les étapes à désactiver en fonction des transitions franchissables (mais pas encore de changement des sorties)
- activation de toutes les étapes à activer, toujours en fonction des transitions franchissables (ceci permet de respecter la règle 5 du Grafcet).



CAPTEURS :	ACTIONNEURS :
hA haut A	MA montée A
bA bas A	DA descente A
hB haut B	RA rotation A
bB bas B	DB descente B
pA présence pièce A	MB montée B
pB présence pièce B	RB rotation B
dcy départ cycle	

Seconde partie, si vous avez facilement traité la première partie :

Programmez, avec la même méthode, le remplissage de bidons (problème A du poly : remplir / boucher / enfoncer). Vous pourrez trouver un document explicatif sur moodle (vers la fin). Testez quand des bidons arrivent en continu, puis quand il en manque un, ou plusieurs.

Déroulement du TP :

faire un rapport de TP à rendre avant une semaine (donc le 11 ou 12 décembre suivant le groupe), voire même dès la fin du TP. Une fois résolue une question, mettre par écrit votre programme, ainsi qu'une rapide présentation des essais que vous avez effectués, et vos éventuelles remarques.