

LAB 8

But du LAB : comment résoudre des problèmes d'automatisation industrielle par un système embarqué à base de μC en utilisant l'algorithmique classique qu'on a utilisé dans les LABs précédents.

Est-ce que l'algorithmique classique est un concept adapté à 100% pour faire de l'automatisation ?

Si oui, pourquoi les automaticiens préfèrent d'autre concept à base des langages adaptés à 100% avec l'automatisation comme le LADDER et le GRAFCET ? Même, il existe carrément des plateformes matérielles, des outils de développement et de simulation dédiés à l'automatisation !

Si non, pourquoi on a arrivé à résoudre le LAB3 par un algorithme qui présente la conversion d'un LADDER en assembleur ?

Ni oui, ni non, c'est quoi cette histoire ?

Réellement, c'est non, elle n'est pas adaptée à 100% pour l'automatisation, par ce que l'algorithme utilisé dans le LAB3 est une conversion d'un concept déjà fait par le LADDER.

Vous pouvez utilisés la conversion du LADDER et GRAFCET en algorithme pour ne pas gaspiller d'une part (l'automatisation des processus de petite taille) et d'autre part de ne pas changer vos concept en tant que automaticien !!!

Donc, pourquoi on casse nos têtes ???

Par ce que :

- Est-ce que tous les automaticiens connaissent comment convertir un LADDER/GRAFCET en algorithme ? Absolument, non !
- D'autre part, tout le monde fait un peu d'algorithmique et de programmation. Donc, on essaye d'exploiter nos connaissances d'algorithmique dans l'automatisation.
- D'autre part, on essaye d'exploiter les μ C dans ce domaine.
- De connaître et d'utiliser le LADDER/GRAFCET/API/Algorithme/ μ C mieux que LADDER/GRAFCET/API seulement... Etc.

Apparemment, il y a pas mal de raisons pour utiliser l'algorithmique dans l'automatisation même elle n'est pas adapté à 100% !

=====

Maintenant en revient à notre LAB :

Le LAB demande d'automatiser le processus de perçage des pièces par le biais d'une perceuse verticale.

La perceuse se compose par :

- Un moteur bidirectionnel (moteur à système vis-écrou) pour déplacer la tête de perçage soit vers le bas soit vers le haut (**MH** et **MB**).
- Un moteur unidirectionnel pour tourner la mèche de perçage (**MP**).
- Deux capteurs fin de course indiquent la position de la tête (en haut (**CH**) ou en bas (**CB**)).
- Un bouton poussoir pour démarrer le cycle de perçage (**START**).
- Un interrupteur pour la mise en marche de processus en plus pour faire un arrêt d'urgence au moment de sa ouverture (**Marche**).

Cahier des charges ou mode de fonctionnement :

- **La mise à l'état initial** : au moment de la fermeture de l'interrupteur « Marche », l'automate fait la vérification du capteur position haute « CH », s'il n'est pas actif l'automate donne l'ordre au moteur de la tête de perçage pour aller vers le haut jusqu'à la position haute.
- **Cycle de perçage** : l'automate attend l'appui sur le bouton « START », si le cas le moteur de perçage est en marche et la tête de perçage descend vers le bas jusqu'au capteur position basse « CB », si ce dernier est actif, la tête de perçage monte vers le haut jusqu'au capteur position haute « CH » et l'automate attend une autre fois l'appui sur le bouton « START » pour démarrer un autre cycle.
- **Arrêt d'urgence** : si l'automate détecte l'ouverture de l'interrupteur « Marche » alors tous les moteurs sont à l'arrêt.

L'étude de n'importe quel processus se commence par un bilan. Ce bilan présente une étude préliminaire en termes de nombre E/S et leurs types.

Le tableau suivant présente le bilan de notre processus de perçage :

Entrées		Sorties	
Nom	Type	Nom	Type
CH	T/R	MH	T/R
CB	T/R	MB	T/R
START	T/R	MP	T/R
MARCHE	T/R (URG)		

Légende : T/R : Tout ou rien. URG : Urgence

La deuxième étape : c'est de faire la liaison entre les E/S de processus avec les pattes de notre μC .

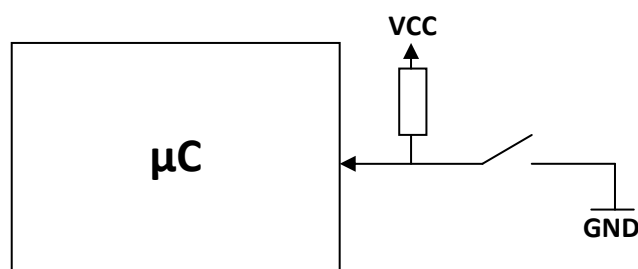
Attention : l'entrée d'urgence présente l'entrée de la plus haute priorité, donc il faut relier en premier lieu ; en plus, dans la plus part des temps, c'est une entrée de demande d'interruption.

Selon la mécanique de l'organe utilisé, on va décider comment relier et comment gérer la demande d'arrêt d'urgence.

Dans notre cas, l'entrée d'arrêt d'urgence est assurée par un interrupteur à deux états stables, l'état ouvert et l'état fermé.

Il existe deux montages :

Montage par PULLUP

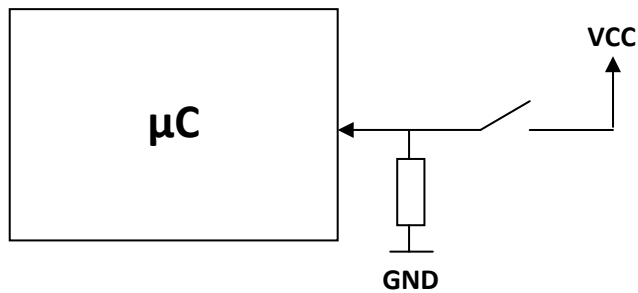


Si l'interrupteur est ouvert, le µC lit un niveau haut à cause de PULLUP.

Si l'interrupteur est fermé, le µC lit un niveau bas.

Donc, au moment d'ouverture de l'interrupteur après fermeture, ce dernier génère un franc montant (RISING-EDGE).

Montage par PULLDOWN



Si l'interrupteur est ouvert, le μC lit un niveau bas à cause de PULLDOWN.

Si l'interrupteur est fermé, le μC lit un niveau haut.

Donc, au moment d'ouverture de l'interrupteur après fermeture, ce dernier génère un franc descendant (FALLING-EDGE).

Dans notre μC , il existe une patte comporte une logique de détection des francs, plus la présence d'une PULLUP (elle suffi d'activer), c'est la patte **RBO**.

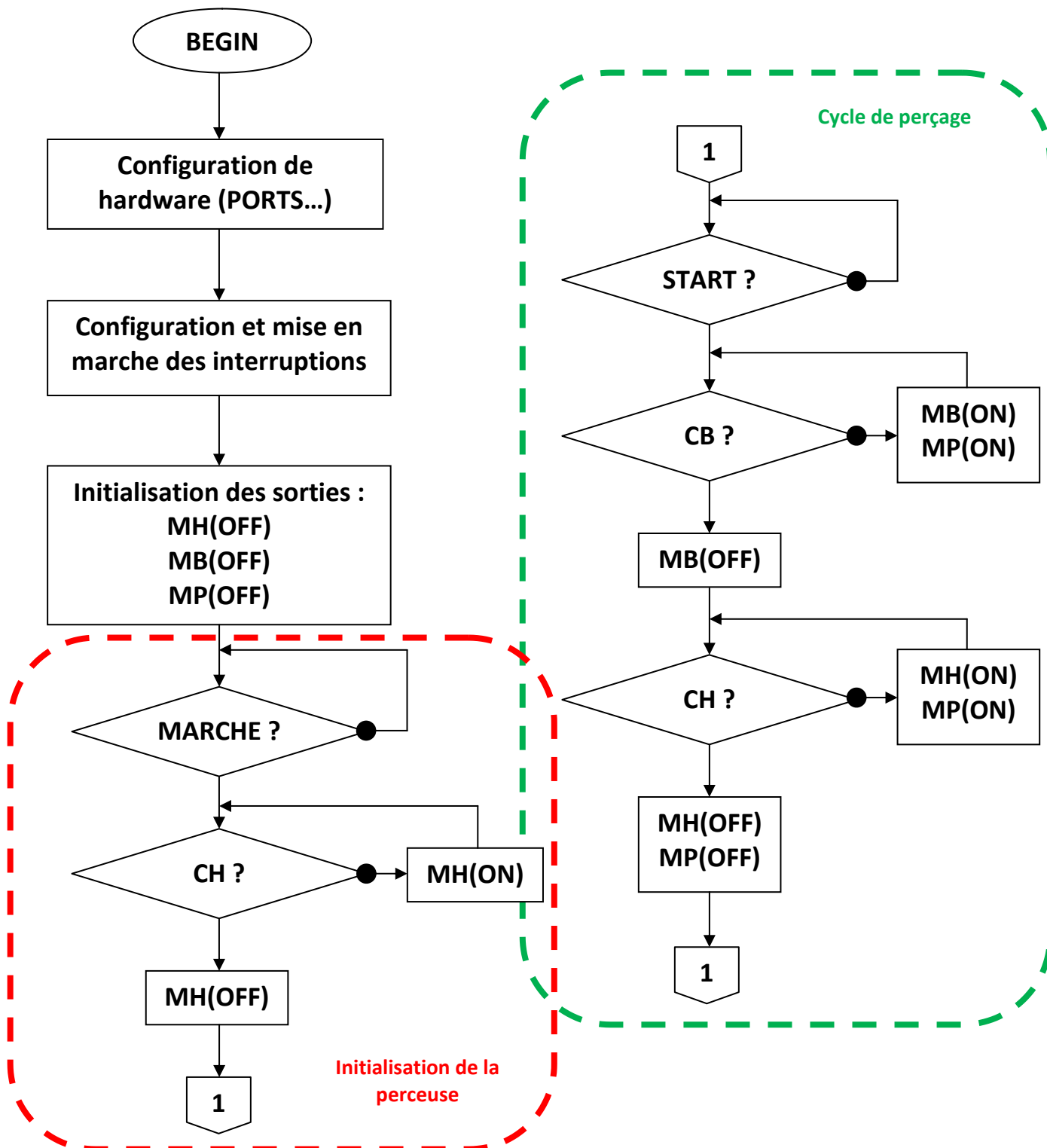
Le reste des E/S sont des T/R simples. On peut relier à n'importe quelle patte de notre μC sauf la patte RA4 par ce qu'elle est drain ouvert et on préfère d'utiliser comme entrée.

Le tableau suivant présente la liaison entre les E/S et les pattes de μC :

Entrées		Sorties	
Processus	μC	Processus	μC
CH	RB3	MH	RA0
CB	RB2	MB	RA1
START	RB1	MP	RA2
MARCHE*	RB0/INT		

Légende : * : entrée d'urgence.

La troisième étape : c'est de faire l'algorithme ou l'organigramme qui suit le cahier des charges.



Traitement de l'arrêt d'urgence se fait sous interruptions (l'arrêt d'urgence signifie un dégât matériel ou/et humain ; dans cette situation l'automate plante sur l'arrêt de tout genre d'actionneur dans le processus).

