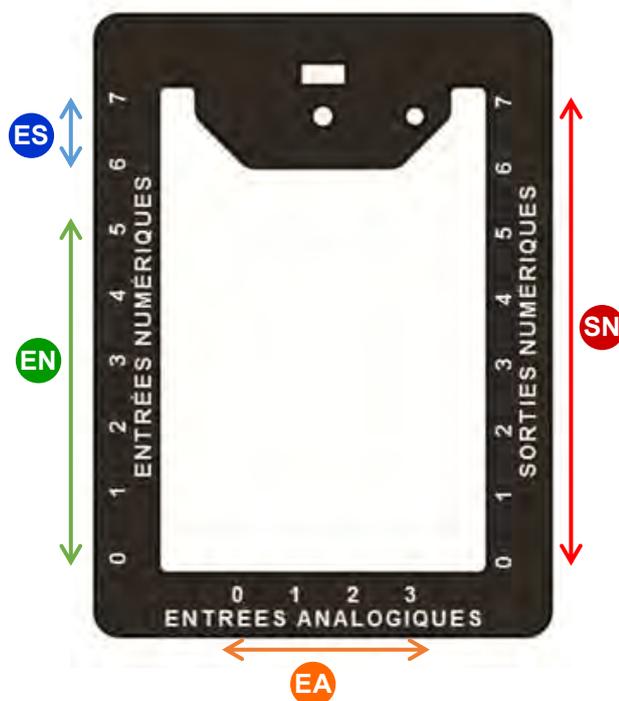


Mise en œuvre du Monte-Charge

Tableau d'affectation des entrées et sorties

ES	Module de communication pour entrées / sorties numériques	Broche Blockly	Etiquette Blockly
7	Communication Bluetooth envoi de données	C.7	BLTH_TX
6	Communication Bluetooth réception de données	C.6	BLTH_RX
EN	Modules capteurs pour entrées numériques		
5	(libre)	C.5	
4	Capteur infrarouge pour télécommande (option)	C.4	Recepteur_IR*
3	Bouton poussoir 1 ^{er} étage	C.3	BP_Etage_1
2	Capteur de fin de course de montée du monte-charge	C.2	FDC_Haut
1	Capteur de fin de course de descente du monte-charge	C.1	FDC_Bas
0	Bouton poussoir rez-de-chaussée	C.0	BP_Etage_0
EA	Modules capteurs pour entrées analogiques		
3	(libre)	A.3	
2	(libre)	A.2	
1	(libre)	A.1	
0	(libre)	A.0	
SN	Modules actionneurs sorties numériques		
7	Connecté à la broche MOTA-2 de la carte contrôle moteur	B.7	Moteur_A2
6	Connecté à la broche MOTA-1 de la carte contrôle moteur	B.6	Moteur_A1
5	(libre)	B.5	
4	(libre)	B.4	
3	(libre)	B.3	
2	(libre)	B.2	
1	Module signal LED rouge 1 ^{er} étage	B.1	LED_Etage_1
0	Module signal LED rouge rez-de-chaussée	B.0	LED_Etage_0



Niveau 2

Exercice niveau : ouverture/fermeture entre fins de course

Objectif : monter et descendre le monte-charge avec 2 secondes d'attente entre chaque mouvement.

Utiliser les capteurs fins de course pour contrôler l'ouverture et la fermeture.

Notions abordées : utilisation des fins de course, procédures (sous-fonctions)



Réf. K-AP-MMR

Correction :

Blocs

```
graph TD
    Start([début]) --> Loop[répéter indéfiniment]
    Loop --> Do[faire]
    Do --> CallUp[appeler sous-fonction monter]
    CallUp --> WaitUp[attendre pendant 2000 ms]
    WaitUp --> CallDown[appeler sous-fonction descendre]
    CallDown --> WaitDown[attendre pendant 2000 ms]
    WaitDown --> Loop
    
    subgraph "sous-fonction monter"
        M1Up[Moteur_A1 désactivée]
        M2Up[Moteur_A2 activée]
        WaitUpSub[attendre jusqu'à entrée FDC_Haut est activée]
        M1Down[Moteur_A1 désactivée]
        M2Down[Moteur_A2 désactivée]
    end
    
    subgraph "sous-fonction descendre"
        M1DownSub[Moteur_A1 activée]
        M2DownSub[Moteur_A2 désactivée]
        WaitDownSub[attendre jusqu'à entrée FDC_Bas est activée]
        M1UpSub[Moteur_A1 désactivée]
        M2UpSub[Moteur_A2 désactivée]
    end
```

Fichier Blockly : MC_N2_A1.xml

Remarque : l'utilisation des sous-fonctions « monter » et « descendre » facilite la lecture du programme.

Exercice niveau 2 : Contrôle de l'ouverture et de la fermeture

Objectif : montée du monte-charge à l'appui du bouton étage 1. Descente du monte-charge à l'appui du bouton étage 0.

Notions abordées : réutilisation des sous-fonctions créées pour un autre programme.

Correction :

Blocs

```
graph TD
    Start[début] --> Loop[répéter indéfiniment]
    Loop --> Wait1[attendre jusqu'à entrée BP_Etage_1 est activée]
    Wait1 --> CallUp[appeler sous-fonction monter]
    CallUp --> Wait0[attendre jusqu'à entrée BP_Etage_0 est activée]
    Wait0 --> CallDown[appeler sous-fonction descendre]
    
    subgraph "sous-fonction monter"
        M1Off[sortie Moteur_A1 désactivée]
        M2On[sortie Moteur_A2 activée]
        FDC_Haut[attendre jusqu'à entrée FDC_Haut est activée]
        M1Off2[sortie Moteur_A1 désactivée]
        M2Off[sortie Moteur_A2 désactivée]
    end
    
    subgraph "sous-fonction descendre"
        M1On[sortie Moteur_A1 activée]
        M2Off2[sortie Moteur_A2 désactivée]
        FDC_Bas[attendre jusqu'à entrée FDC_Bas est activée]
        M1Off3[sortie Moteur_A1 désactivée]
        M2Off3[sortie Moteur_A2 désactivée]
    end
```

Fichier Blockly : MC_N2_A2.xml

Exercice niveau 2 - A.3 : Contrôle ouverture/fermeture avec BP et signal d'arrivée

Objectif : Faire monter et descendre le monte-charge à l'aide des boutons-poussoirs sans distinction, faire en sorte qu'une LED clignote lors d'une manœuvre de la barrière.

Notions abordées : utilisation d'opérateur logique OU (+)

Correction :

Blocs

The code is organized into three main parts:

- Main Program:**
 - début**
 - répéter indéfiniment** (infinite loop):
 - faire** (do):
 - répéter** (repeat):
 - fixer test_bouton à** (set test button to): `entrée BP_Etage_1` or `entrée BP_Etage_0`
 - jusqu'à test_bouton = 1** (until test button = 1)
 - fixer test_bouton à 0** (set test button to 0)
 - si entrée FDC_Bas est activée** (if FDC_Bas input is active):
 - faire** (do): `appeler sous-fonction monter`
 - sinon** (else): `appeler sous-fonction descendre`

- sous-fonction monter** (v1.3.1):
- sortie Moteur_A1** désactivée
- sortie Moteur_A2** activée
- tant que entrée FDC_Haut est désactivée** (while FDC_Haut input is deactivated):
 - faire** (do):
 - `basculer LED_Etage_1`
 - attendre pendant 100 ms** (wait 100 ms)
- sortie Moteur_A1** désactivée
- sortie Moteur_A2** désactivée
- sortie LED_Etage_1** désactivée
- sous-fonction descendre**:
- sortie Moteur_A1** activée
- sortie Moteur_A2** désactivée
- tant que entrée FDC_Bas est désactivée** (while FDC_Bas input is deactivated):
 - faire** (do):
 - `basculer LED_Etage_0`
 - attendre pendant 100 ms** (wait 100 ms)
- sortie Moteur_A1** désactivée
- sortie Moteur_A2** désactivée
- sortie LED_Etage_0** désactivée

Fichier Blockly : MC_N2_A3.xml

Exercice niveau 2 - A.4 : Contrôle des LED lors d'une entrée dans une nouvelle boucle

Objectif : Reprendre l'exercice précédent, une LED doit rester allumée à l'étage où se trouve le monte-charge

Remarque : Utiliser les capteurs fin de course et des conditions

Correction :

Blocs

```
graph TD
    Start[début] --> Loop1[répéter indéfiniment]
    Loop1 --> Loop2[faire]
    Loop2 --> Fix1[fixer test_bouton à 1]
    Loop2 --> Fix2[fixer test_bouton à 0]
    Loop2 --> Cond1[si entrée FDC_Bas est activée]
    Cond1 --> Call1[appeler sous-fonction monter]
    Cond1 --> Call2[appeler sous-fonction descendre]
    Loop2 --> End1[fin]

    subgraph monter [sous-fonction monter]
        M1[Moteur_A1 désactivée]
        M2[Moteur_A2 activée]
        Loop3[tant que entrée FDC_Haut est désactivée]
        Loop3 --> Toggle1[basculer LED_Etage_1]
        Toggle1 --> Cond2[si entrée FDC_Bas est activée]
        Cond2 --> LED0[LED_Etage_0 activée]
        Cond2 --> LED0[LED_Etage_0 désactivée]
        Loop3 --> Wait1[attendre pendant 100 ms]
        Wait1 --> M1
        Wait1 --> M2
        Wait1 --> LED1[LED_Etage_1 activée]
    end

    subgraph descendre [sous-fonction descendre]
        M3[Moteur_A1 activée]
        M4[Moteur_A2 désactivée]
        Loop4[tant que entrée FDC_Bas est désactivée]
        Loop4 --> Toggle2[basculer LED_Etage_0]
        Toggle2 --> Cond3[si entrée FDC_Haut est activée]
        Cond3 --> LED1[LED_Etage_1 activée]
        Cond3 --> LED1[LED_Etage_1 désactivée]
        Loop4 --> Wait2[attendre pendant 100 ms]
        Wait2 --> M3
        Wait2 --> M4
        Wait2 --> LED0[LED_Etage_0 activée]
    end
```

Fichier Blockly : MC_N2_A4.xml